–Building Your Own– GLOBE Weather Station

An Technical Instruction Manual for High School STEM Curriculum

Chitanya Gopu, David Martinez, Hayley Walker, Michelle Ye

TABLE OF CONTENTS

1 INTRODUCTION

1.1 WEATHER, GLOBE, AND CITIZEN SCIENCE (WHY A WEATHER STATION?)

The Earth's atmosphere is changing rapidly. To accurately track weather patterns over time and to meaningfully understand the environment, U.S. National Aeronautics and Space Administration (NASA) and the National Science Foundation (NSF) sponsors the Global Learning and Observations to Benefit the Environment (GLOBE) program to collect weather data worldwide. Not only do scientists contribute data to NASA GLOBE's program, but average citizens and students can participate as well to encourage "citizen science", educating the public while accomplishing GLOBE's main mission. See

For relevant earth science background, as well as some instrumentation details, it is recommended you check out these links provided by GLOBE:

For the general GLOBE site:
https://www.globe.gov/

For Temperature:
https://www.globe.gov/documents/348614/93d4bb3c-79e3-4255-9fc8-537fc4f870dc

For Pressure:
https://www.globe.gov/documents/348614/a1ab9ecb-f8b1-41ba-9dd9-a420895f954b

For Humidity:
https://www.globe.gov/documents/348614/89f8c44d-4a99-494b-ba81-1853b80710b4

1.2 HOW TO USE THIS MANUAL

This manual is split into six sections. Sections 2 to 5 describe the different components of your built weather station, while the sixth is an optional component to replace Section 4 for a more in-depth mechanical engineering design curriculum. The manual describes all tools, materials, costs, and procedures to build your GLOBE weather station, as well as instructions for Wi-Fi data logging. Ratings, info boxes, and debugging tips are included as needed for clarity. While this manual is a beta version, we sincerely hope it is clear and easy to follow.

1.3 RATINGS AND INFO BOXES EXPLAINED

Due to the DIY (Do It Yourself) nature of the project, many construction and hardware procedures have multiple options, or ways of achieving the same result. Additionally, any materials or components in the BOM can be replaced if your high school already has materials that would work. Because of this, when options are given in the manual, a rating system will be used to rank the relative cost, engineering complexity, and material/tool availability of each option. The ratings are as follows:

Cost:

$	$ $	$ $ $
Very cheap ($0.00 to $5.00 each). You can buy several backups if you are afraid of messing up.	Affordable, or in the middle ($5.00 - $20.00).	Expensive! ($21.00+) Be careful with this component. Cheaper alternatives will usually be given as well.

Complexity:

?

Very easy to understand and build. This is intuitive or just very simple and will not take very much time.

? ?

In the middle. Very understandable if you read the instructions carefully. May take a bit longer or be more involved to build.

? ? ?

A fun challenge! Learn some real engineering skills. Looking at additional resources given in this manual will aid understanding, and instructor's help is recommended.

Material/Tool requirements:

✂

Not material intensive. Tools needed include office supplies and/or your hands.

✂ ✂

May involve tools you haven't used before, but are fairly common, easily learned, and reasonably safe with the right technique: hammers, screwdrivers, allen wrenches, hand drills, or electronic components that can heat up.

✂ ✂ ✂

Check your school's resources before jumping into this option. May require supervision and help. These tools require more involved safety procedures, or are not as easily obtained. This included tap drills, soldering irons, or machine shop clamps. There are usually easier alternatives given.

In addition to these ratings, sections with TIPS, COMPARE RESULTS (which will give the results we got when we built this weather station for comparison) and MORE INFORMATION (for external links for further research) are scattered throughout this manual.

1.4 TOOLS

The tools you'll need for Sections 1 to 5:

- ☐ Soldering iron and solder
- ☐ Phillips Screwdriver
- ☐ Hand drill and drill bits
- ☐ Table saw
- ☐ Ruler (or calipers if available)
- ☐ Sharpie

1.5 MATERIALS (BOMs)

BOM 1 (Required Purchases)

BOM #	Item Common Name	Listed Name (ID)	Vendor	Quantity	Cost	Purchase Link
1	Power Boost	PowerBoost 1000 Basic - 5V USB Boost @ 1000mA from 1.8V+ (ID 2030)	Adafruit	1	$14.95	https://www.adafruit.com/products/2030
2	Solar Panel Battery	Lithium Ion Polymer Battery - 3.7v 2500mAh (ID 328)	Adafruit	1	$14.95	https://www.adafruit.com/products/328
3	Weather Sensor	Adafruit BME280 12C or SPI Temperature Humidity Pressure Sensor (ID 2652)	Adafruit	1	$19.95	https://www.adafruit.com/products/2652
4	Wifi Microcontroller	Adafruit HUZZAH ESP8266 Breakout (ID 2471)	Adafruit	1	$9.95	https://www.adafruit.com/products/2471
5	Breadboard	Medium BreadBoard (ID 65)	Adafruit	1	$5.00	https://www.adafruit.com/products/65
6	Radiation Shield	AcuRite 06054M Temperature & Humidity Solar Radiation Shield	Amazon	1	$20.70	https://tinyurl.com/mg5e8eg
7	Adaptor Cable	3.5 / 1.3mm or 3.8 / 1.1mm to 5.5 / 2.1mm DC Jack Adapter Cable (ID 2788)	Adafruit	1	$0.95	https://www.adafruit.com/products/2788
8	USB Cable	USB to TTL Serial Cable - Debug / Console Cable for Raspberry Pi (ID 954)	Adafruit	1	$9.95	https://www.adafruit.com/products/954
9	JST Cable	JST 2-pin cable (ID 261)	Adafruit	2	$0.75	https://www.adafruit.com/product/261
10	Anemometer	Anemometer (ID 1733)	Adafruit	1	$54.06	https://www.adafruit.com/products/1733
11	Battery Charger	USB / DC / Solar Lithium Ion/Polymer charger (ID 390)	Adafruit	1	$17.50	https://www.adafruit.com/products/390
12	Solar Panel	Medium 6V 2W Solar panel (ID 200)	Adafruit	1	$29.00	https://www.adafruit.com/products/200
13	HDPE Plastic	HDPE (marine grade)	McMaster-Carr	1	$16.61	https://www.mcmaster.com/#9785t212/=17eh6o5

14	Tripod	60-Inch Lightweight Tripod	Amazon	1	$23.50	https://tinyurl.com/l77mdn2
15	Heat Gun	Darice Multi-Purpose Heat Tool, 320 Watt	Walmart	1	$11.35	https://tinyurl.com/mvspvmx
26	Solderable Breadboard	Sparkfun Solder-able Breadboard	Sparkfun	1	$4.95	https://www.sparkfun.com/products/12070

Total Cost: $249.17 (Allow for shipping costs to your area).

Item 16 is optional for a sturdier electronics system.

Screws and Nuts BOM (SN BOM)

It is recommended that you get screws and nuts at your local hardware store; online, these come in packs of 100, which is many more than necessary. Item #3 is optional, but will make creating threads easier.

Item #	Item	Link
1	1/4-20 x 3/4" Steel Phillips Rounded Head Screws (100 pack for $9.92)	https://www.mcmaster.com/#90272a540/=17eiqrz
2	4-24 x 1/2" Thread-Cutting Screws for Plastic (100 pack for $4.66)	https://www.mcmaster.com/#94629a560/=17eiwuv
3	4-40 x 1" Steel Phillips Rounded Head Screws (100 pack for $3.00)	https://www.mcmaster.com/#90272a115/=17eizw5
4	1/4-20 Medium-Strength Steel Hex Nut (100 pack for $4.40)	https://www.mcmaster.com/#95462a029/=17ej2xc
5	4-40 Stainless Steel Hex Nut (100 pack for $2.61)	https://www.mcmaster.com/#91841a005/=17ej4wi

BOM 2 (for Section 5 use only)

BOM #	Item Common Name	Listed Name (ID)	Vendor	Quantity	Cost	Purchase Link
17	PVC pipe and fittings	1.5" PVC pipe (1), 1 ft long, tee (2), 1 ½" to ¾" bushings (3), cap (1) (Home Depot)	Home Depot	-	$13.18	Buy from your local hardware store. We went to Home Depot
18	Small PVC Pipe	Thick Walled Unthreaded PVC Pipe	Home Depot	1	$4.68	https://www.mcmaster.com/#48855k22/=1762zcn (Recommended: buy at hardware store instead)
19	PVC Cement and Primer	PVC Cement and Primer	Amazon	1	$9.10	http://preview.tinyurl.com/lgoxy8p
20	Aluminum Shaft	Anodized Aluminum rotary shaft	McMaster-Carr	1	$11.83	https://www.mcmaster.com/#5911k11/=17eh288
22	Photo interrupter	1/8" Gap Photo Interrupter Slotted Optical Switch	Amazon	1	3.83	https://tinyurl.com/lzyurbj
23	Shaft Clamp	Clamping Shaft Collar	McMaster-Carr	2	$4.40	https://www.mcmaster.com/#6157k12/=17h2u9p
25	Coffee Scoops!	Chef Craft 4 Piece Perfect Coffee Measuring Spoon Scoop, Black	Amazon	1 package	$6.80	https://tinyurl.com/l6r2dga
26	Ball Bearing	R4A-2RS Bearing 1/4"x3/4"x9/32" inch Sealed Miniature	Amazon	1	$5.37	https://tinyurl.com/lfabtbu

27	Conduit Clamp	2-in. PVC conduit clamp OR 2-in. Rigid 2-hole strap	McMaster-Carr	2	$2.50	https://tinyurl.com/mob7hom
28	Super Glue	Super Glue	Amazon	1 package	$1.88	https://tinyurl.com/mk7v95y
29	Screw Item 29	6-32 x 3/8″ Phillips or Allen cap screw an nuts	Local Hardware Store	3	-	-
30	Screw Item 30	3-48 x 3/8″ set screw or Phillips screw	Local Hardware Store	1	-	-
31	O-rings	¼″ O-rings	Local Hardware Store	2	-	-
32	Handheld Anemometer	GM816 Digital Anemometer	Walmart	1	$15.97	https://tinyurl.com/lokntax

Total Cost: $44.68 (Without Allow for shipping costs to your area).

Item 32 is optional to test for your handmade anemometer’s accuracy.

1.6 SYSTEM OVERVIEW

The weather station you will build is relatively cheap, solar powered, and wirelessly plots data in real-time on an online platform called ThingSpeak, using an Arduino (see other sections for details). The system measures temperature, humidity, pressure and wind speed using a breakout board and anemometer, with a custom designed electronics housing and a roof mount.

This project has three components; electrical, mechanical, and data receiving (Wi-Fi).

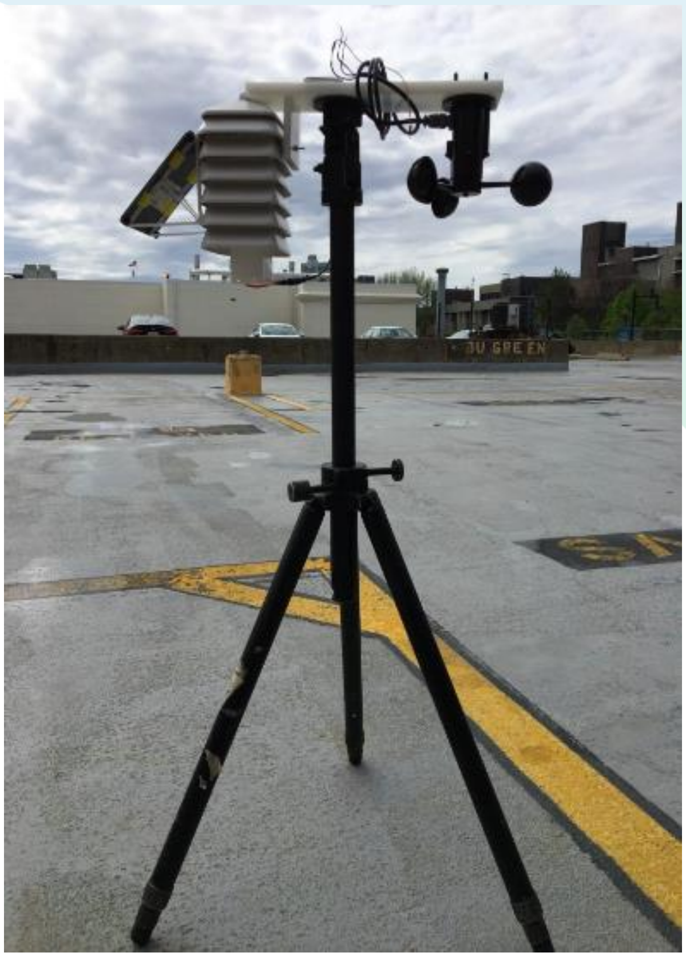


Fig. 1. The weather station on our roof.

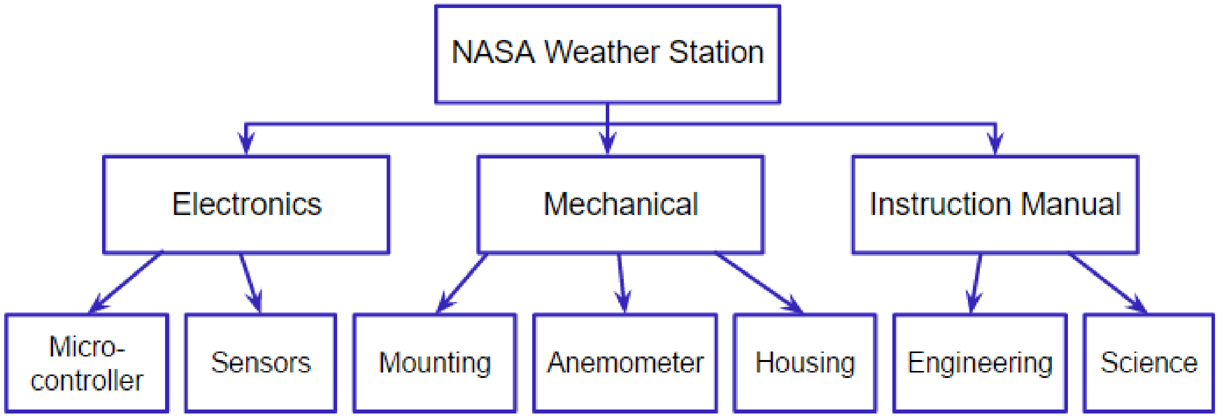


Fig. 2. Functions of the weather station kit.

1. The electrical portion includes the microcontroller and weather sensors. Weather data is collected by the sensors by telling the sensors what to do with Arduino code, then stored in the Wi-Fi micro-controller.
2. The mechanical components of the weather station kit include the housing and mounting. The housing shields the sensors and electronics from rain, and outdoor conditions, while keeping them inside the radiation shield. The mounting holds the electrical components to the tripod, and the tripod to the roof.
3. The data collected by the sensors will then be sent to your computer, to ThingSpeak (known as the User Interface, or UI) computer.

If this sounds complicated, don't worry! This manual is detailed and informative. You'll learn skills like how microcontrollers work, beginner Arduino coding, basic breadboard wiring and connections, how sensors work, how to take accurate measurements, how to build structures from real engineering drawings, and how to use tools safely. Have fun!

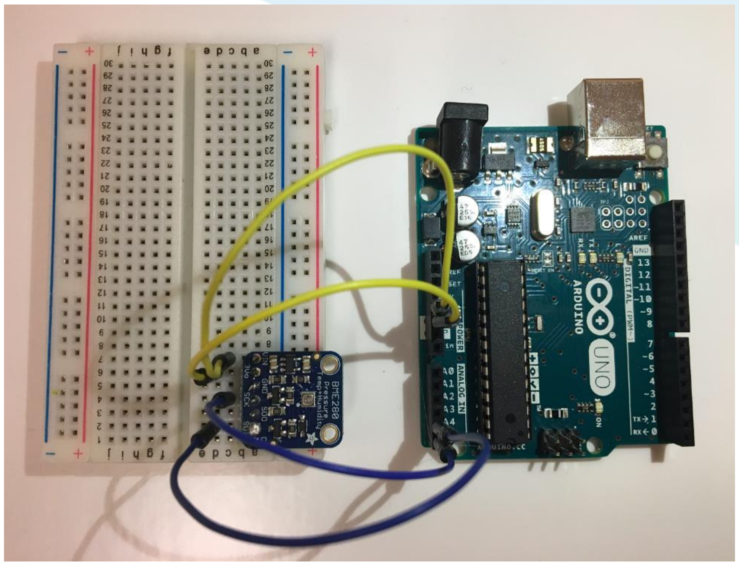
2 SENSORS AND WIFI

This section is split into 3 parts. The first part is an Arduino exercise connecting the BME280 weather sensor to an Arduino, and outputting results onto the serial monitor. The Second section is an exercise with the ESP8266 Wi-Fi microcontroller. The third section integrates the sensor and ESP8266, along with all other electrical components, to make up the total hardware and software components of the weather station.

1. A. Sensor Wiring - HARDWARE

Use this wiring if you want to connect via I2C interface.

- Put the breakout board on the breadboard vertically so the pins are in rows 1-7 in column a (without covering any other labelled column).
- Use a wire with one end in row 7, column c (Vin) to the arduino pin labeled 3.3V.
- Use a wire with one end on row 5, column c (GROUND) and the other end to GND on the arduino in the power section.
- Use a wire with one end on row 4, column c (SCK) with the other end on A5 (analog section) on the Arduino
- Use a wire with one end on row 2, column c (SDI) to A4 (analog section) on the Arduino



B. SOFTWARE:

Copy this code to Arduino IDE to collect data. Once the script is compiled and uploaded, open the serial monitor to see the data.

```
#include <Wire.h>
#include <SPI.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>

#define BME_SCK 13
#define BME_MISO 12
#define BME_MOSI 11
#define BME_CS 10

#define SEALEVELPRESSURE_HPA (1013.25)

Adafruit_BME280 bme; // I2C
//Adafruit_BME280 bme(BME_CS); // hardware SPI
//Adafruit_BME280 bme(BME_CS, BME_MOSI, BME_MISO, BME_SCK); //
software SPI

unsigned long delayTime;

void setup() {
   Serial.begin(9600);
   Serial.println(F("BME280 test"));
   bool status;

   // default settings
   status = bme.begin();
   if (!status) {
       Serial.println("Could not find a valid BME280 sensor, check 
wiring!");
       while (1);
   }

   Serial.println("-- Default Test --");
   delayTime = 1000;

   Serial.println();
}


void loop() {
   printValues();
   delay(delayTime);
}


void printValues() {
   Serial.print("Temperature = ");
   Serial.print(bme.readTemperature());
   Serial.println(" *C");

   Serial.print("Pressure = ");

   Serial.print(bme.readPressure() / 100.0F);
   Serial.println(" hPa");

   Serial.print("Approx. Altitude = ");
   Serial.print(bme.readAltitude(SEALEVELPRESSURE_HPA));
   Serial.println(" m");

   Serial.print("Humidity = ");
   Serial.print(bme.readHumidity());
   Serial.println(" %");

   Serial.println();
                                   }
```

CHALLENGE

Now we will have a challenge! We've been taking data in celsius, but we usually think in Fahrenheit! So how can we get the data to display in Fahrenheit?

See if you can find where to put that conversion in the code. The equation to convert from Celsius to Fahrenheit is:

$$T(F)=T(C)*1.8+32$$

2. Wi-fi – HARDWARE

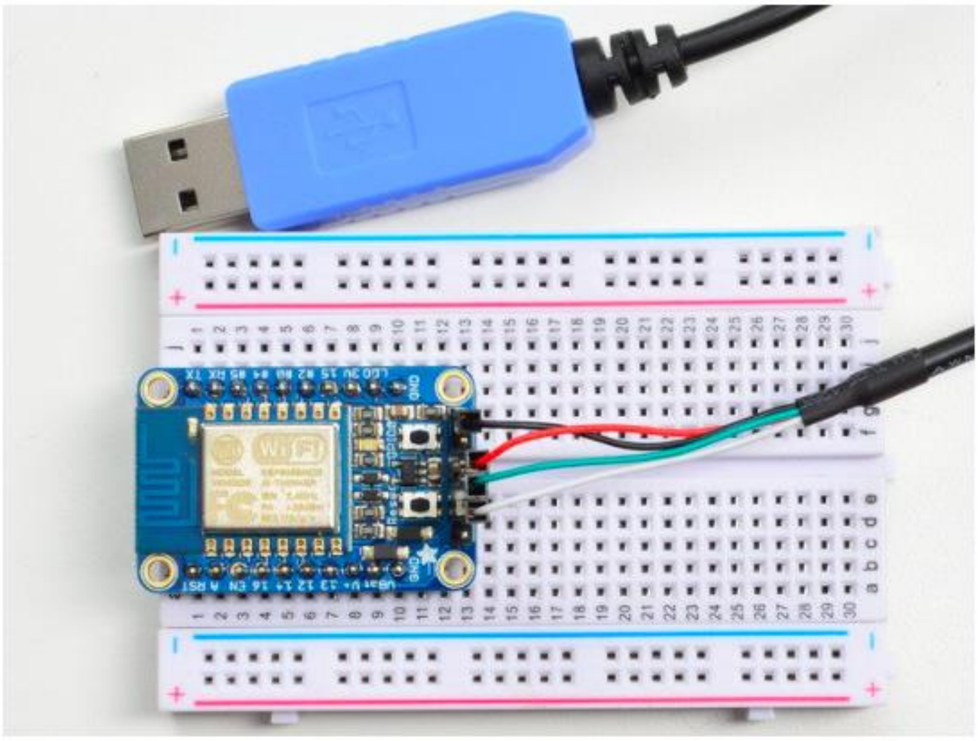
- Please download the following: the ESP8266 Board Package, USB driver, Arduino IDE 1.6.4.
- Download the ESP8266 Huzzah Wifi files → https://learn.adafruit.com/adafruit-huzzah-esp8266-breakout/using-arduino-ide#install-the-arduino-ide-1-dot-6-4-or-greater
- Follow the steps from "Install the ESP8266 Board Package" to "Setup ESP8266 Support"

Setting up your computer for the wifi component:

- Open the arduino IDE program

Go to Tools → Board → Adafruit Huzzah ESP8266 → Port → "\dev\cu.SLAB_USBtoUART"

If using a console cable, connect the black wire to ground, red wire to **V+**, white wire to **TX** and green wire to **RX**



You will see the red and blue onboard LED flicker when powered up, but they will not stay lit.

Blink Test

This test is a simple code to make sure the wifi board and microcontroller are functioning properly. It will make the build in LED on the microcontroller blink every half a second!

Enter this into the sketch window (and save since you'll have to)

```
void setup() {
pinMode(0, OUTPUT);
}

void loop() {
digitalWrite(0, HIGH);
delay(500);    //delay time is in milliseconds
digitalWrite(0, LOW);
delay(500);
}
```

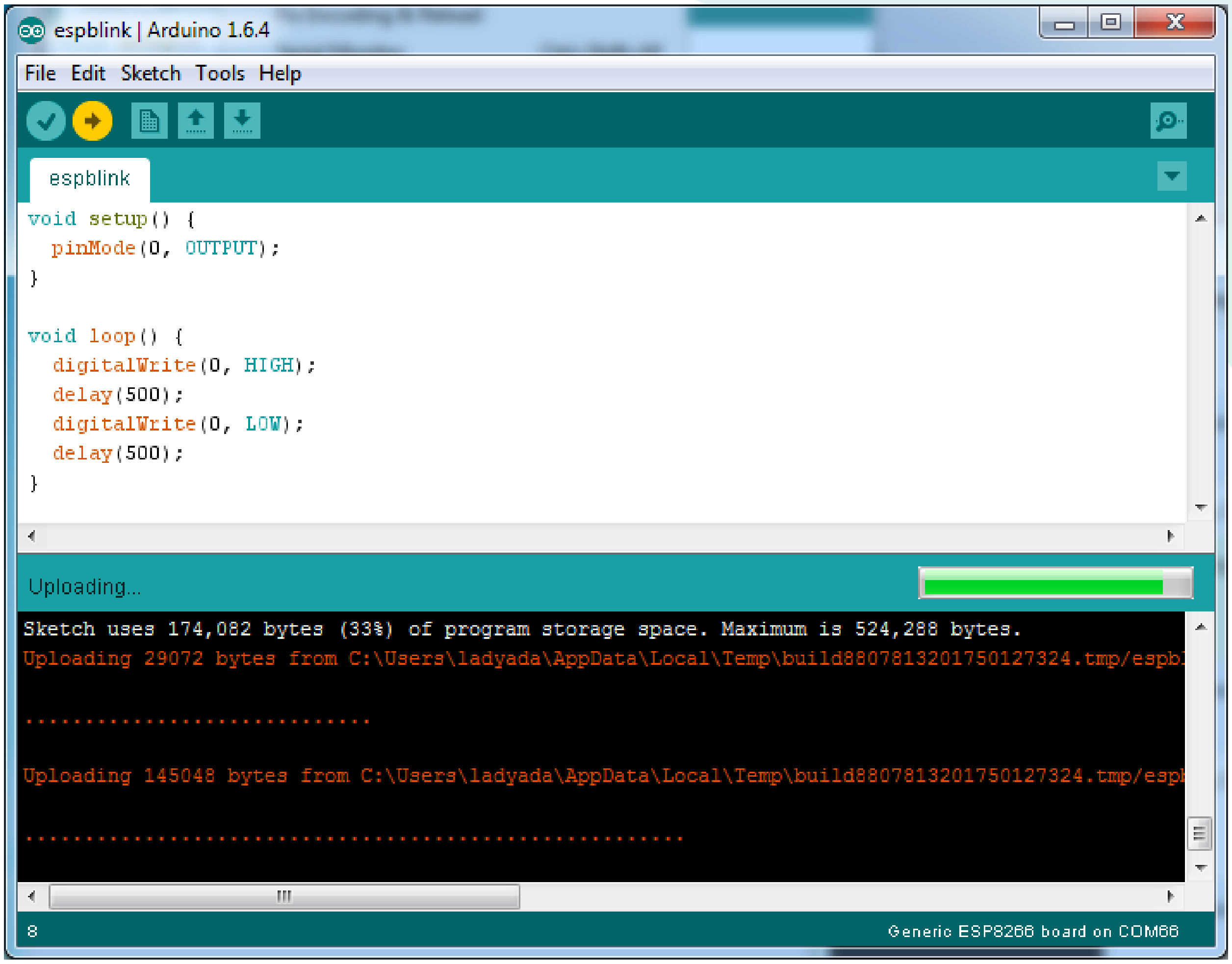
Now you'll need to put the board into bootload mode, to set up the arduino before running code. You'll have to do this before each upload. There is no timeout for bootload mode, so you don't have to rush!

1. Hold down the GPIO0 button, the red LED will be lit
2. While holding down GPIO0, click the RESET button
3. Release RESET, then release GPIO0
4. When you release the RESET button, the red LED will be lit dimly, this means it's ready to bootload

Once the ESP board is in bootload mode, you are almost ready to run the code.

Before the script can be uploaded, click verify to compile the sketch and make sure the code is written properly.

Then, click on the 'Upload' button. The second button from the left on the toolbar.



```
void setup() {
  pinMode(0, OUTPUT);
}

void loop() {
  digitalWrite(0, HIGH);
  delay(500);
  digitalWrite(0, LOW);
  delay(500);
}
```

```
Sketch uses 174,082 bytes (33%) of program storage space. Maximum is 524,288 bytes.
Uploading 29072 bytes from C:\Users\ladyada\AppData\Local\Temp\build8807813201750127324.tmp/espbl

.............................

Uploading 145048 bytes from C:\Users\ladyada\AppData\Local\Temp\build8807813201750127324.tmp/esph

.......................................................
```

Supplemental Information (explanation of code):

First, we have the 'setup' function. This is run when the reset button is pressed. It is also run whenever the board resets for any reason, such as power first being applied to it, or after a sketch has been uploaded.

In this case, there is just one command there, which, as the comment states tells the Arduino board that we are going to use the pin 0 as an output.

It is also mandatory for a sketch to have a 'loop' function.

Unlike the 'setup' function that only runs once, after a reset, the 'loop' function will, after it has finished running its commands, immediately start again.

Once you've run the code, go back and change with the numbers to adjust the speed of the blinking. For example:

```
void loop() {
digitalWrite(0, HIGH);   // turn the LED on (HIGH is the voltage level)
delay(1000);               // wait for a second
digitalWrite(0, LOW);    // turn the LED off by making the voltage LOW
delay(1000);               // wait for a second
}
```

Inside the loop function, the commands first of all turn the LED pin on (HIGH), then 'delay' for 1000 milliseconds (1 second), then turn the LED pin off and pause for another second.

The sketch will start immediately - you'll see the LED blinking. Hooray!

Connecting via WiFi

Once you've got the LED blinking, let's go straight to the fun part, connecting to a web server. Create a new sketch with this code:

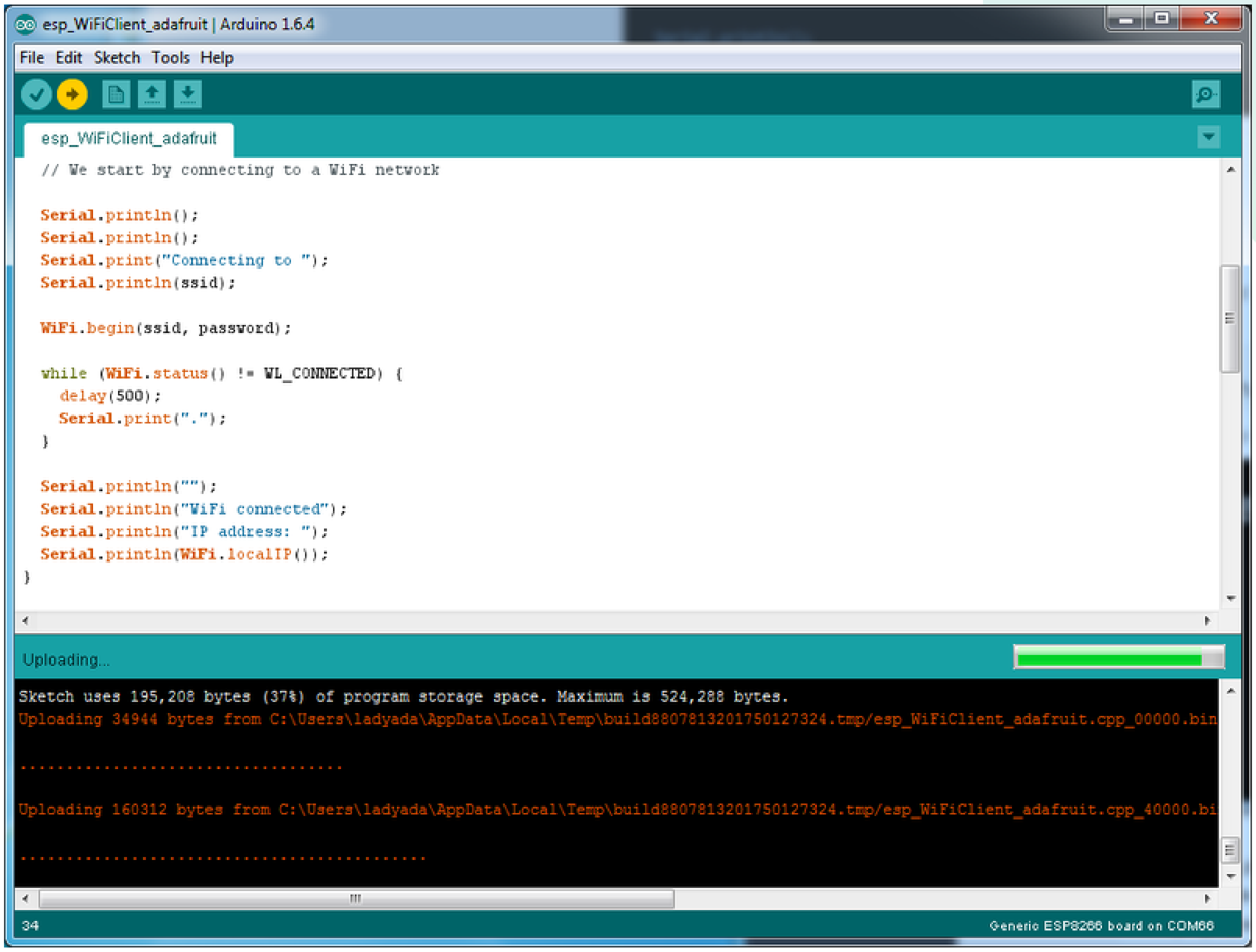
```
/*
*  Simple HTTP get webclient test
*/

#include <ESP8266WiFi.h>

const char* ssid     = "yourssid";
const char* password = "yourpassword";

const char* host = "wifitest.adafruit.com";

void setup() {
Serial.begin(115200);
delay(100);

// We start by connecting to a WiFi network

Serial.println();
Serial.println();
Serial.print("Connecting to ");
Serial.println(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
}

Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
}

int value = 0;

void loop() {
delay(5000);
++value;

Serial.print("connecting to ");
Serial.println(host);

// Use WiFiClient class to create TCP connections
WiFiClient client;
const int httpPort = 80;
if (!client.connect(host, httpPort)) {
  Serial.println("connection failed");
  return;
}

// We now create a URI for the request
String url = "/testwifi/index.html";
Serial.print("Requesting URL: ");
Serial.println(url);

// This will send the request to the server
client.print(String("GET ") + url + " HTTP/1.1\r\n" +
             "Host: " + host + "\r\n" +
             "Connection: close\r\n\r\n");
delay(500);

// Read all the lines of the reply from server and print them to Serial
while(client.available()){
  String line = client.readStringUntil('\r');
  Serial.print(line);
}

Serial.println();
Serial.println("closing connection");
}
```

Don't forget to update

```
const char* ssid     = "yourssid";
const char* password = "yourpassword";
```

to your wifi network and password, then upload the same way: get into bootload mode, then upload code via IDE.

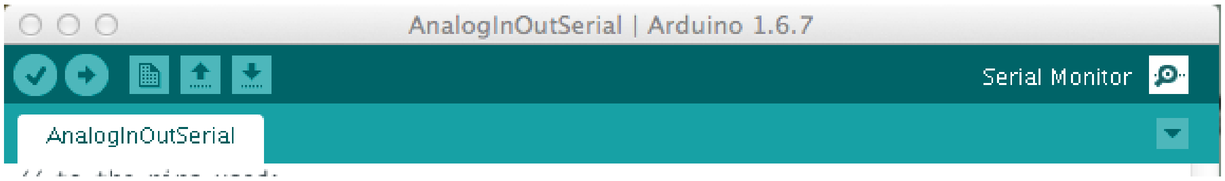
NOTE: Code only works for single password servers.

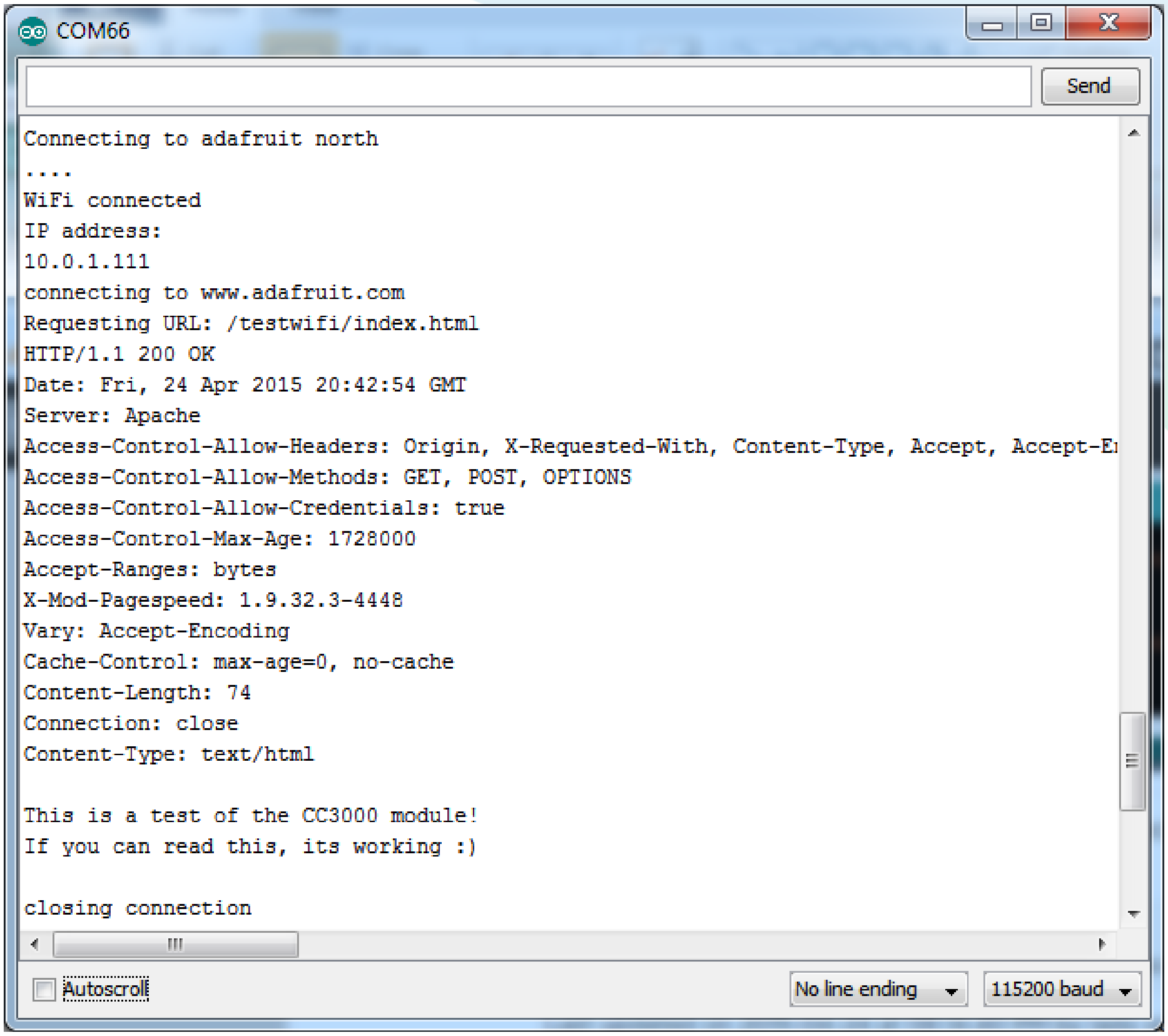

esp_WiFiClient_adafruit | Arduino 1.6.4
File Edit Sketch Tools Help
esp_WiFiClient_adafruit
Uploading...
34
Generic ESP8266 board on COM66

```
  // We start by connecting to a WiFi network

  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

```
Sketch uses 195,208 bytes (37%) of program storage space. Maximum is 524,288 bytes.
Uploading 34944 bytes from C:\Users\ladyada\AppData\Local\Temp\build8807813201750127324.tmp/esp_WiFiClient_adafruit.cpp_00000.bin

..................................

Uploading 160312 bytes from C:\Users\ladyada\AppData\Local\Temp\build8807813201750127324.tmp/esp_WiFiClient_adafruit.cpp_40000.bi

............................................
```

Open up the IDE serial console at 115200 baud to see the connection and webpage printout! You can check the baud at the bottom of the serial monitor (the drop down menu).

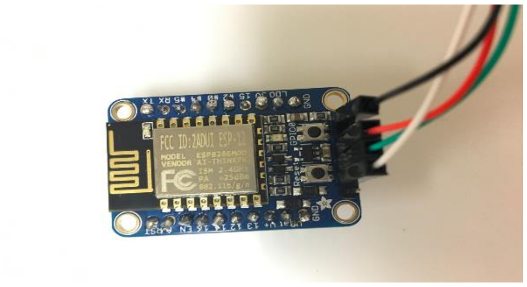




```
Connecting to adafruit north
....
WiFi connected
IP address:
10.0.1.111
connecting to www.adafruit.com
Requesting URL: /testwifi/index.html
HTTP/1.1 200 OK
Date: Fri, 24 Apr 2015 20:42:54 GMT
Server: Apache
Access-Control-Allow-Headers: Origin, X-Requested-With, Content-Type, Accept, Accept-E
Access-Control-Allow-Methods: GET, POST, OPTIONS
Access-Control-Allow-Credentials: true
Access-Control-Max-Age: 1728000
Accept-Ranges: bytes
X-Mod-Pagespeed: 1.9.32.3-4448
Vary: Accept-Encoding
Cache-Control: max-age=0, no-cache
Content-Length: 74
Connection: close
Content-Type: text/html

This is a test of the CC3000 module!
If you can read this, its working :)

closing connection
```

That's it, pretty easy!

This page was just to get you started and test out your module. For more information, check out the ESP8266 port github repository for much more up-to-date documentation!



3. ENTIRE ELECTRICAL ASSEMBLY (HARDWARE AND SOFTWARE)

This section first covers how to solder, a necessary skill involved in this assembly.

What is soldering?

Soldering is when any of various alloys is fused and applied to the joint between metal objects to unite them without heating the objects to the melting point.

If you're new to soldering - check out this link: http://www.wikihow.com/Solder-Electronics

Now that you're an expert and you've read all the information on soldering → Proceed!

1. Make sure to solder all the electrical components individually first; follow the instruction given for each component
    a. ESP8266 Huzzah: https://learn.adafruit.com/adafruit-huzzah-esp8266-breakout/assembly
    b. Power Boost: https://learn.adafruit.com/adafruit-powerboost-1000-basic/overview
    c. BME280: https://learn.adafruit.com/adafruit-bme280-humidity-barometric-pressure-temperature-sensor-breakout/assembly

d. Solar Charger: The first thing to do before starting to charge with a solar panel is to solder the large filtering capacitor. Solder the longer leg on the capacitor into the + end on the solar charger within a white circle. Then solder the shorter leg into the - end on the solar charger within the white circle.

2. Solder pin wires to the end of each cable anemometer
3. Solder two ends of the red JST wires onto two adjacent pins on the switch
4. Then solder the two ends of the JST black wires to create a double JST wire

Wifi to Sensor Connections:

1. Connect the two red (+) columns with a pin wire
2. Connect the two blue (-) columns with a pin wire
3. Put the BME 280 breakout board on the breadboard vertically so the pins are in rows 16-22 in column a (without covering any other labelled column).
4. Use a pin wire from column j, row 1 to the blue (-) column and a pin wire on column j, row 2 to the red (+)
5. Put ESP Huzzah 8266 on the breadboard with the two rows of pins on rows 1-10 (with the GND pins on row 1, and the extra pins overhanging the board) with one row of pins on column i and the other on column c.
6. Use a pin wire with one end on row 22, column c (Vin) on breakout board to row 3, column a (3V) on microcontroller
7. Use a pin wire with one end on row 20, column e (Gnd) on breakout board to row 1, column b (GND) on microcontroller
8. Use a pin wire with one end on row 19, column c (SCK) on breakout board to row 8, column a (#5) on microcontroller
9. Use a pin wire with one end on row 17, column c (SDI) on breakout board to row 7, column a (#4) on microcontroller

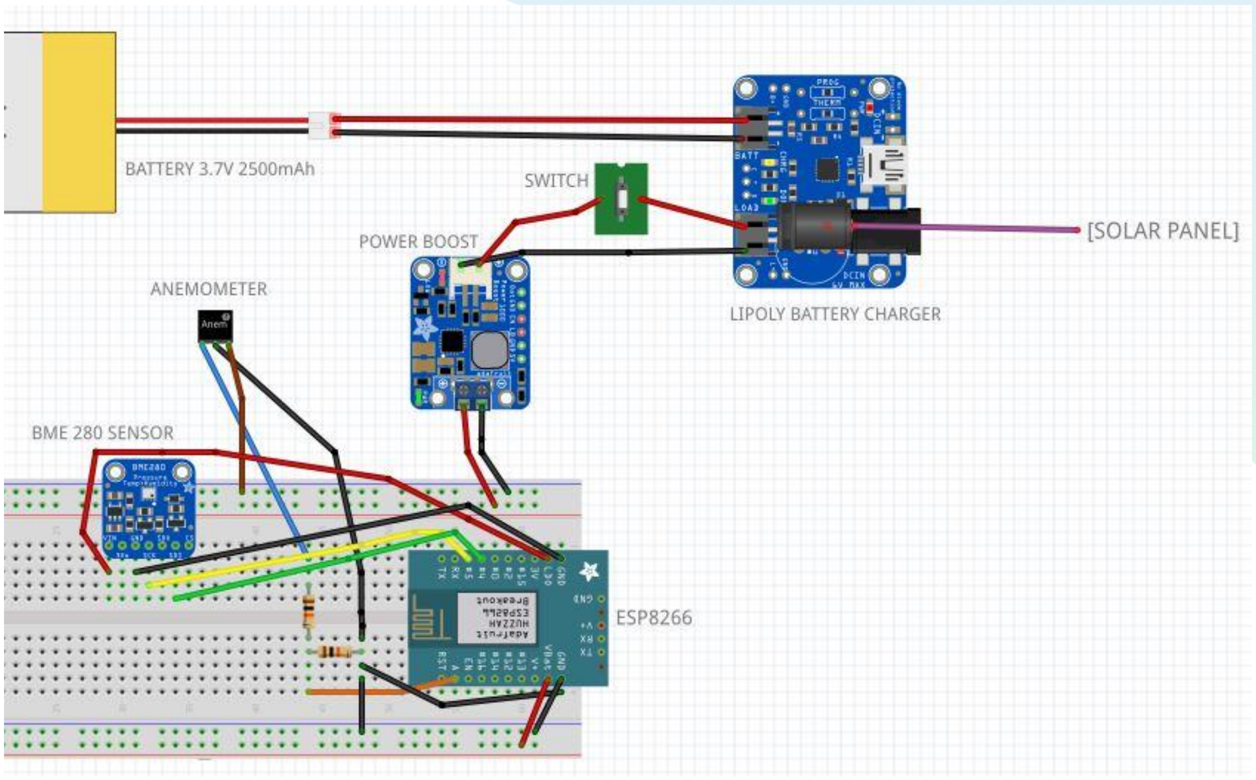
Anemometer and Voltage Divider:

1. Plug the blue anemometer pin wire into column b, row 28
2. Put a 10K resistor with one end on column d, row 28 to column f, row 28
3. Use another 10K resistor with one end on column i, row 28 to column j, row 29
4. Use a pin wire on column i, row 28 to column j, row 29
5. Connect a pin wire from column g, row 20 to the blue (-) column
6. Place one end of a pin wire column i, row 20 and solder it to the black anemometer pin wire
7. Plug the brown anemometer pin wire to the red (+)

Other connections:

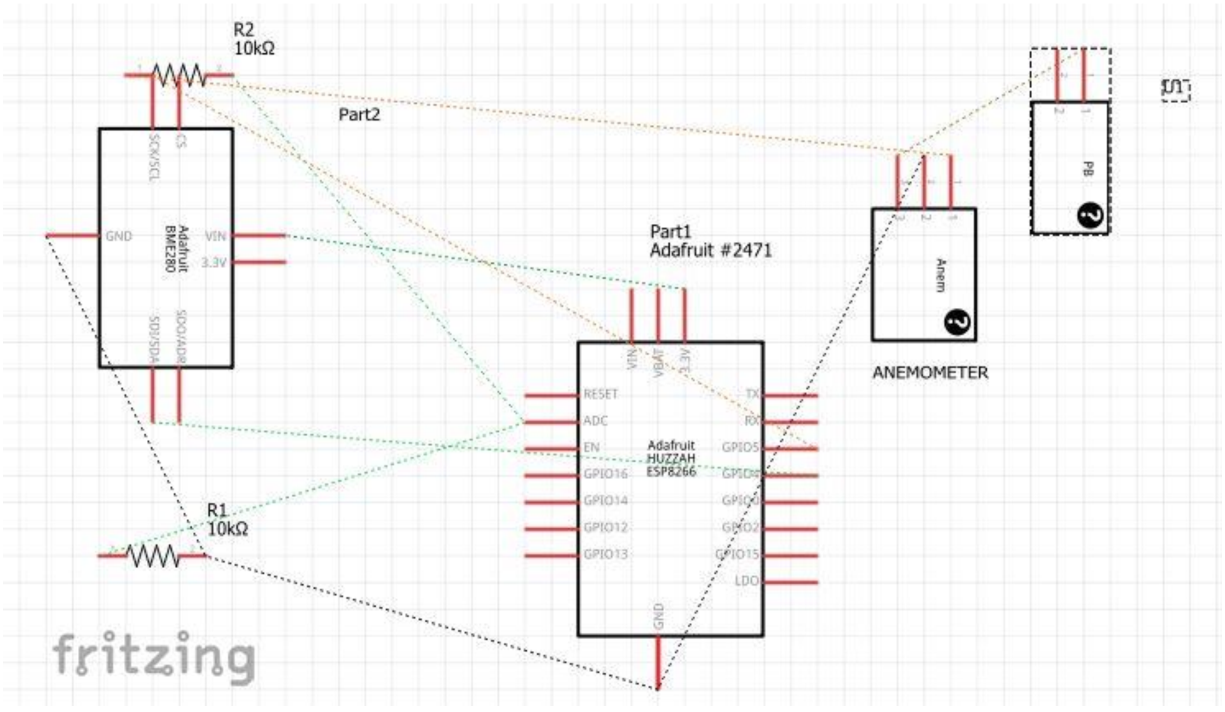
Connecting the Solar panel to the rest of the electronics

1. Take the 3.5 / 1.3mm or 3.8 / 1.1mm to 5.5 / 2.1mm DC Jack Adapter Cable and plug one end into the solar panel and the other end to the appropriate port on the Lipoly battery charger.
2. Then connect the 3.7V 2500mAh battery into the JST port labeled "BATT" on the Lipoly Battery charger.
3. Then take a double ended jst cable (with a soldered switch) and connect one end to the "LOAD" port and then the other end to the JST port on the Power Boost.
4. Then connect two pin cables with the other side of the Power Boost, the side closest to the power button.
5. Take the pin pin wires connected to the - side on the Power boost and connect it to the blue (-) column.
6. Take the pin pin wires connected to the + side on the Power boost and connect to the red (+).



Schematic of all electrical components.

WHEN CONNECTED - THERE SHOULD A BLUE FLASH FROM THE WIFI MICROCONTROLLER TO INDICATE THAT THE BATTERY IS CHARGED.



Specs sheets and more information about the sensors can be found using the links in BOM 1. Additionally, Adafruit and Sparkfun have many resources for electronics information:

https://www.adafruit.com/

https://www.sparkfun.com/

This link explains some basic circuit theory:

https://learn.sparkfun.com/tutorials/what-is-a-circuit

Voltage Divider:

- Since the anemometer outputs voltages from 0-2V, but the ESP8266 only accepts inputs of 0-1V, the anemometer output needs to be reduced.
- A voltage divider is a simple circuit that turns a large voltage into a smaller one.
    - This is done using two resistors, in series, between the input (the anemometer) and output (ESP 8266) voltage of the circuit.

Software:

First, the class must create a ThingSpeak account. ThingSpeak is a data analytics software that is used to receive the data your school collects in real time, and graphs it on multiple channels. Make one channel for each measurable (temperature, pressure, humidity, and wind speed). A link to ThingSpeak is below:

https://thingspeak.com/

Copy-Paste the following code into the Arduino IDE terminal to get your system working. Check the Serial Monitor to see the output values in real time and to make sure its working.

- Update the yellow highlighted code sections with your school's Wi-fi channel and password.

- Update the green highlighted code with your channel ID name and API key for each measurable when you set up a ThingSpeak account.

```
/*
   Thingspeak API Upload for BME 280 and Anemometer
*/
#include <Wire.h>
#include <SPI.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>
#include <ESP8266WiFi.h>
#include <ThingSpeak.h>

#define BME_SCK 13
#define BME_MISO 12
#define BME_MOSI 11
#define BME_CS 10

#define SEALEVELPRESSURE_HPA (1013.25)
Adafruit_BME280 bme;
unsigned long delayTime;

#include <ESP8266WiFi.h>

const int analogInPin = A0;
int sensorValue = 0;
float sensorVoltage = 0.0;
float Windspeed = 0;

//Wifi Host
const char* ssid     = "David's iPhone";
const char* password = "t72xk9toe178w";

WiFiClient client;

//Server Information
const char* host = "api.thingspeak.com";
const int postingInterval = 30 * 1000; // post data every 30
seconds to prevent missing points

// ThingSpeak Temperature Channel
const int TempchannelID = 252947;
String TempwriteAPIKey = "8MQNBPETGJFHQTET"; // write API key
for your ThingSpeak Channel

// ThingSpeak Humidity Channel
```

```
const int HumchannelID = 252949;
String HumwriteAPIKey = "C7TM5S2GLDSG3YV6"; // write API key
for your ThingSpeak Channel

// ThingSpeak Pressure Channel
const int PrechannelID = 255500;
String PrewriteAPIKey = "LIOXGIYPYUOR3VSU"; // write API key
for your ThingSpeak Channel

// ThingSpeak Wind Channel
const int WindchannelID = 265952;
String WindwriteAPIKey = "PYE9PHFT8BYUQT0I"; // write API key
for your ThingSpeak Channel

void setup() {
 Serial.begin(115200);
 delay(100);
 Serial.println(F("BME280 test + Wifi + Thingspeak"));

 bool status;

 // default settings
 status = bme.begin();
 if (!status) {
   Serial.println("Could not find a valid BME280 sensor, check
wiring!");
   while (1);
 }

 Serial.println("-- Default Test --");
 delayTime = 5000;

 Serial.println();

 // Now we connect to a WiFi network
 Serial.print("Connecting to ");
 Serial.println(ssid);

 WiFi.begin(ssid, password);

 while (WiFi.status() != WL_CONNECTED) {
   delay(500);
   Serial.print(".");
 }

 Serial.print("");
 Serial.print("WiFi connected");
 Serial.println();
 Serial.print("IP address: ");
 Serial.println();
 Serial.print(WiFi.localIP());
 Serial.println();

 int value = 0;
 delay(5000);
 ++value;

 Serial.print("connecting to ");
 Serial.println(host);
 Serial.println();
}

void loop() {
 if (client.connect(host, 80)) {

   // Measure Signal Strength (RSSI) of Wi-Fi connection-
Collect values
   float rssi = bme.readTemperature();
   float rssi2 = bme.readHumidity();
   float rssi3 = bme.readPressure() / 100;

   // Show values that were measured
   Serial.print('\n');
   Serial.println("Weather Values");
   printValues();
   Wind();
   float rssi4 = Windspeed;

   //Create String using value
   String body = "field1=";
   body += String(rssi, 2);
   //Create String using value
   String body2 = "field1=";
   body2 += String(rssi2, 2);
   //Create String using value
   String body3 = "field1=";
   body3 += String(rssi3, 2);
   String body4 = "field1=";
   body4 += String(rssi4, 2);

   Serial.print("Posting to Thingspeak");
   Serial.println();

   //Post on the Temperature channel
   client.print("POST /update HTTP/1.1\n");
   client.print("Host: api.thingspeak.com\n");
   client.print("Connection: close\n");
   client.print("X-THINGSPEAKAPIKEY: " + TempwriteAPIKey +
"\n");
   client.print("Content-Type: application/x-www-form-
urlencoded\n");
   client.print("Content-Length: ");
   client.print(body.length());
   client.print("\n\n");
   client.print(body);
   client.print("\n\n");



   // Post on Humidity Channel
   client.print("POST /update HTTP/1.1\n");
   client.print("Host: api.thingspeak.com\n");
   client.print("Connection: close\n");
   client.print("X-THINGSPEAKAPIKEY: " + HumwriteAPIKey +
"\n");
   client.print("Content-Type: application/x-www-form-
urlencoded\n");
   client.print("Content-Length: ");
   client.print(body2.length());
   client.print("\n\n");
   client.print(body2);
   client.print("\n\n");



   // Post on Pressure Channel
   client.print("POST /update HTTP/1.1\n");
   client.print("Host: api.thingspeak.com\n");
   client.print("Connection: close\n");
   client.print("X-THINGSPEAKAPIKEY: " + PrewriteAPIKey +
"\n");
   client.print("Content-Type: application/x-www-form-
urlencoded\n");
   client.print("Content-Length: ");
   client.print(body3.length());
   client.print("\n\n");
   client.print(body3);
   client.print("\n\n");


   // Post on Windspeed Channel
   client.print("POST /update HTTP/1.1\n");
   client.print("Host: api.thingspeak.com\n");
   client.print("Connection: close\n");
   client.print("X-THINGSPEAKAPIKEY: " + WindwriteAPIKey +
"\n");
   client.print("Content-Type: application/x-www-form-
urlencoded\n");
   client.print("Content-Length: ");
   client.print(body4.length());
   client.print("\n\n");
   client.print(body4);
   client.print("\n\n");


   delay(postingInterval);

   Serial.println("Done Posting to Thingspeak");
   // wait and then post again

  }
 client.stop();
}

//Windspeed function
void Wind() {
 sensorValue = analogRead(analogInPin);
 sensorVoltage = sensorValue * .00322266; // Convert from
0...1024 to 0...5v
 //Serial.println("Sensor Value: ");
 //Serial.print(sensorValue);
 //Serial.println("Sensor Voltage: ");
 //Serial.print(sensorVoltage);

 Windspeed = ((sensorVoltage - .66) / (.04969136)) * 2.23694;

 if (Windspeed >= 0) {
   Serial.print("WindSpeed = ");
   Serial.print(Windspeed);
   Serial.print(" mph");
   Serial.println();
 }

 if (Windspeed < 0) {
   Serial.print("WindSpeed = ");
   Serial.print(0);
   Windspeed = 0;
   Serial.print(" mph");
   Serial.println();
 }

}
//Sensor Function Outside loop
void printValues() {
 Serial.print("Temperature = ");
 Serial.print(bme.readTemperature());
 Serial.println(" *C");

 Serial.print("Pressure = ");

 Serial.print(bme.readPressure() / 100);
 Serial.println(" hPa");

 Serial.print("Humidity = ");
 Serial.print(bme.readHumidity());
 Serial.println(" %");
}
```

3 ELECTRONICS HOUSING

$? ✂ ✂

This section describes how to make a housing for the electronics. It must be waterproof and include ventilation for the sensors.

3.1 Materials Needed

- ☐ Polly Plastics Heat Moldable Plastic Sheets - 3 Sheets, 8 x 12 x 1/16
  - Alternatives: Waterproof tape on cardboard, Tupperware
- ☐ Heating gun
  - Alternatives: Hot Water (at least 150º F)
- ☐ Shears
  - Alternatives: Gardening Shears, regular scissors
- ☐ Ruler
- ☐ Sharpie
  - Alternatives: Pen
- ☐ Bucket of cold water

The housing will look like this when it's finished, in addition to a lid.

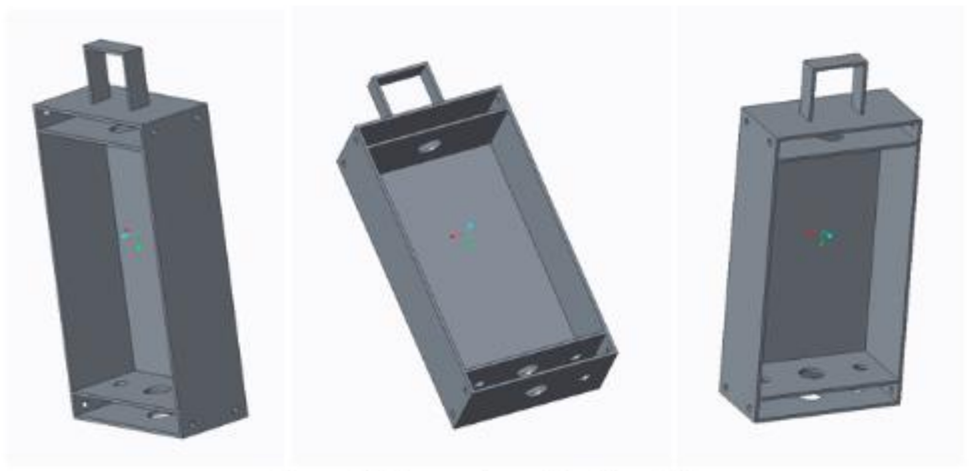


Figure 1: Dimensions: 6" x 3" x 1.5"

Notes: When you cut out the plastic, make sure you are cutting a little more outside the box instead of inside the box to account for more precise pieces! Remember, it is easier to cut out parts than to add in parts.

PART 1: CUTTING OUT THE PIECES

1. Measure out 2 pieces of 6″ by 3″ pieces
   a. Use a ruler & pen to outline the lines
   b. Cut out the pieces
2. Measure out 4 pieces of 3″ by 1.5″ pieces
   a. Use a ruler & pen to outline the lines
   . b. Cut out the pieces
3. Measure out 2 pieces of 6″ by 1.5″ pieces
   . a. Use a ruler & pen to outline lines
   b. Cut out the pieces
10. Measure out 1 piece of 4″ by 0.4″ piece for for the hook
   a. Use a ruler & pen to outline lines
   b. Cut out the pieces

SPOT CHECK: Do you have 8 pieces of plastic?

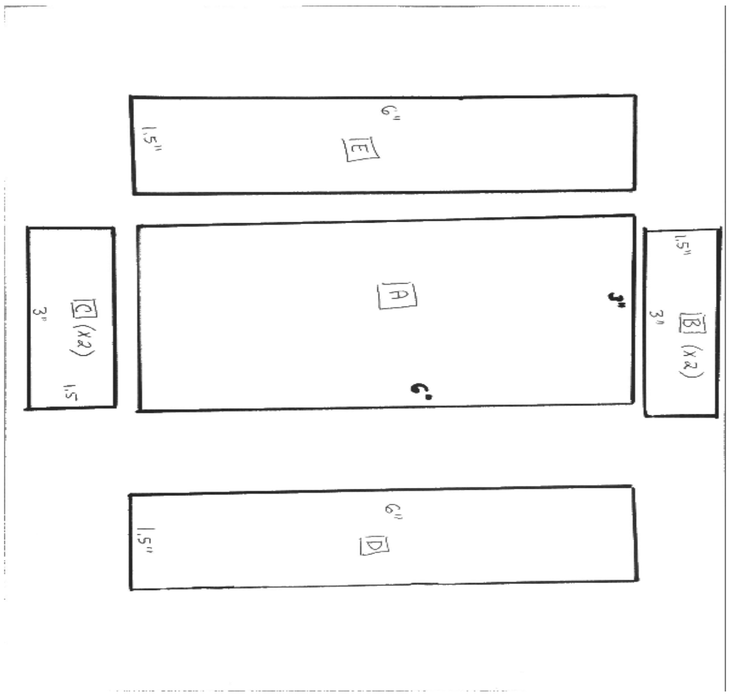
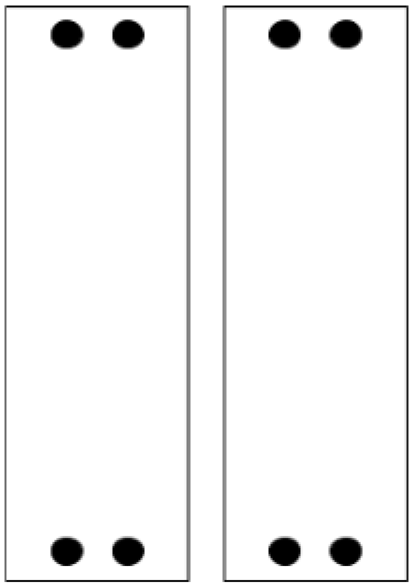


Figure 2: Schematic of the box pieces labeled with corresponding letters, used in Parts 2-4
Note: Pieces B&C need to be done twice so there will be two pieces of B and two piece of C.

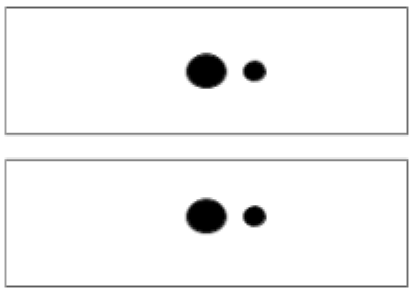
PART 2: DRILLING THE PIECES

This portion requires use of a drill; please check in with a supervisor before drilling.

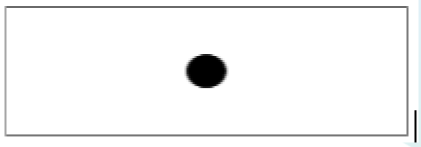
Note: The following schematics are are not drawn to size but instead should be a reference to how you would drill your pieces.



1. Drill two 0.196” holes (using a size 9 drill bit) near the edge of D and E
2. Drill 1 large hole (using several ¼” holes combined with a ¼ drill bit) and 1 ¼” hole in the center of C
    a. Do it for both pieces



3. Drill 1 ¼” hole in the center of B1 (only 1 piece)



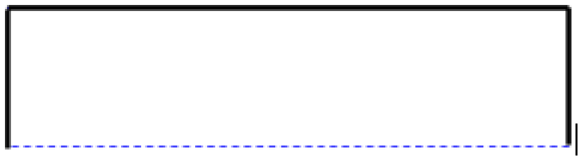
4. Leave B2 (the other piece) untouched

PART 3: ASSEMBLING THE PIECES

Notes: The easiest way would be to assemble the outside layers first and then move to inside layers. The drilled holes are not included for simplicity.

1. Using a heat gun, heat up the edges of D for roughly 5 seconds until the edge of plastic becomes slightly clear

a. Refer to diagram below → Heat up along the dotted blue line

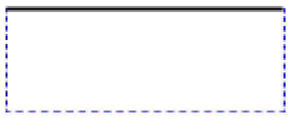


2. Attach the heated up side of D to the long side of A
3. Repeat (3) for the other side of A with piece E
4. Make sure the sides are perpendicular or forms a 90 degree angle
5. Cool in a tub of cold water to let the plastic set again

Spot Check: Does your base have 2 pieces of 6'' x 1.5'' attached perpendicularly to the edges?

6. Using a heat gun, heat up the edges of the C1 for roughly 5 seconds until the edge of the plastic because slightly clear

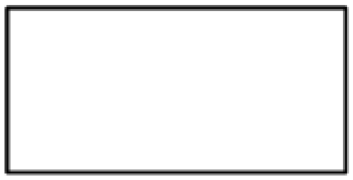
a. *Refer to diagram below* → Heat up along the dotted blue lines and leave 1 long side not heated (labeled in black)



7. Attach C1 to the shorter side of A, effectively making a box
8. Repeat (7) for the other side of the 6x3 piece with B2
9. Cool in a tub of cold water to let the plastic set again

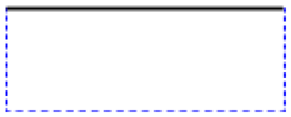
Spot Check: You should have a box that looks like this if you look down at it(look below)

It will be 6'' x 3'' x 1.5''

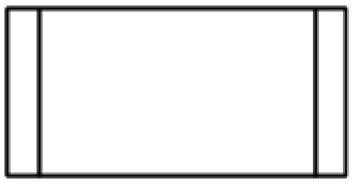


10. Additionally heat up the last 3′′x1.5′′ pieces (C2, B1) like (6)

a. Refer to diagram below - Heat up the dotted blue lines and leave 1 long side not heated (labeled in black)



11. Attach C2 plastic about 0.5 inches from C1 to resemble a diagram like below
12. Attach B1 about 0.5 inches from B2 to resemble a diagram like below



Note: It is easier to attach the longer pieces first because when you put in the 3′′ x 1.5′′ pieces in the inside - it pushes back & expand the box to help keep the structure in place.

Part 4: Attaching the hook to the box

1. Cut out a 0.4" by 4" strip of moldable plastic
2. Heat up the edges of the strip using the heat gun
3. Mold the plastic into a circle
4. Heat up the connected edges and attach that to B2
5. The box is ready to be snapped into the radiation shield now.

Here is the CAD drawing of the assembly:

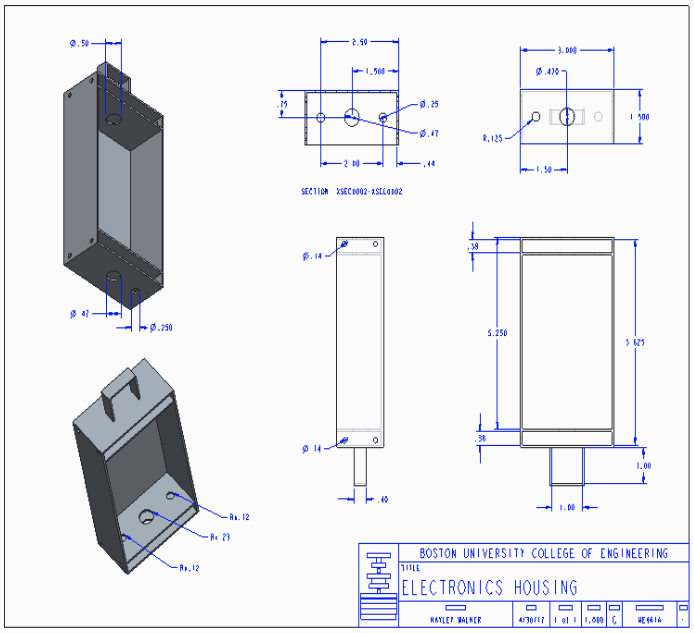
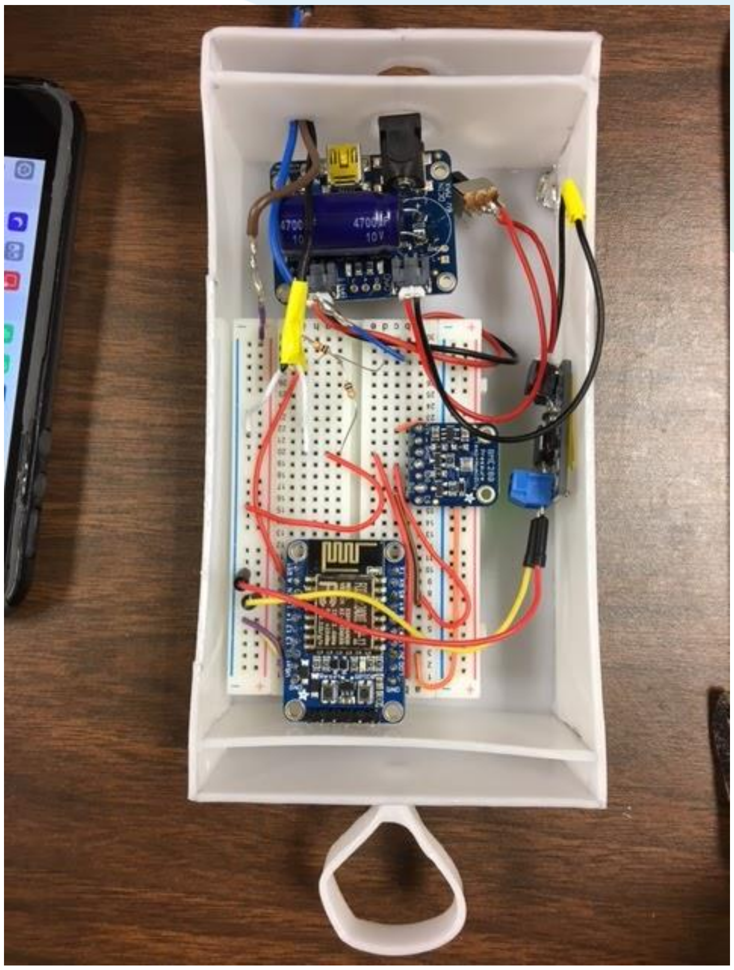


Figure 3. CAD Drawing of the complete housing box including hook.

When the electronics are placed inside the housing, it will look like this:



It is recommended to use Velcro or a non-heating method of attaching the components in the housing. Another suggestion is to place a layer of polystyrene insulation between the breadboard and battery (tucked under the breadboard), to prevent the slight heat from the battery affecting sensor readings.

4 MOUNTING

(to be done by teachers or other professionals)

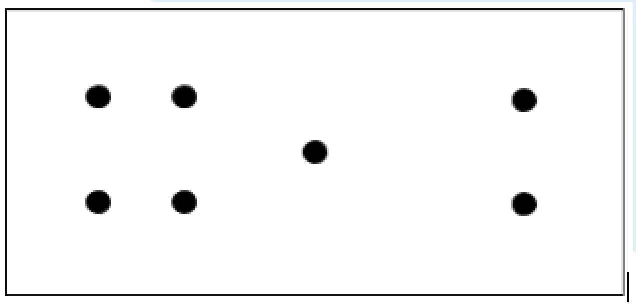
This section outlines how the radiation shield and anemometer will mount to a tripod, and how the tripod will mount to a roof safely and securely.

Materials Needed:

- ☐ Marine- Grade HDPE Plastic
- ☐ BOM Line #1 x 5
- ☐ BOM Line #3 x 3
- ☐ Hand drill
- ☐ Hand saw
- ☐ Tripod
- ☐ Bucket
- ☐ Wet cement or sand

Steps:

1. Use the saw to cut a piece of HDPE plastic to be a 12" by 3" rectangle.
2. Cut another piece of HDPE that is a 3" by 3" square
3. Using the hand drill, drill four holes (¼") on one side of the rectangle 1" away from the edge (to mount the anemometer) through the piece of plastic
4. Drill another hole (¼) into the other side, about 1" deep to hang the radiation shield
5. Then, drill two holes 2" away from that edge, that are 0.65" away from the center of the piece, through the plastic.



6. In the smaller square, drill one hole (4-48) 2″ away from one edge, which has two holes (4-48) into the edge 0.65″ away from the center. Refer to the CAD drawings for more details.
7. If you have tapping equipment available, tap all the holes to fit the appropriate screws. If not simply use the drill to push the screws into the holes to create threads.
8. Attach the anemometer to the four ¼-20 holes using the ¼-20 1″ screws, and place nuts on the other side to secure them
9. Hang the radiation shield off a ¼-20 screw on the other end, (put the screw through the radiation shield hook before screwing it in)
10. Attach the smaller piece to the larger rectangle using the two 4-48 holes, and screwing them together using the 4-40 screws.
11. Put a 4-40 screw through the remaining hole in the smaller square, and then hang the other slit on the radiation shield off that screw
12. Create one final hole in the larger rectangle that is 4.5″ away from the edge with the radiation shield in the center (width measurement) for a ¼-20 screw and tap it using a ¼-20 screw but then remove the screw. This hole will be used to screw the whole assembly to the tripod.
13. In the three buckets, pour either wet cement or sand when the mount is ready.
14. Place each of the three legs into one of the buckets
15. When the cement hardens, or if using sand immediately, the system is ready to be used

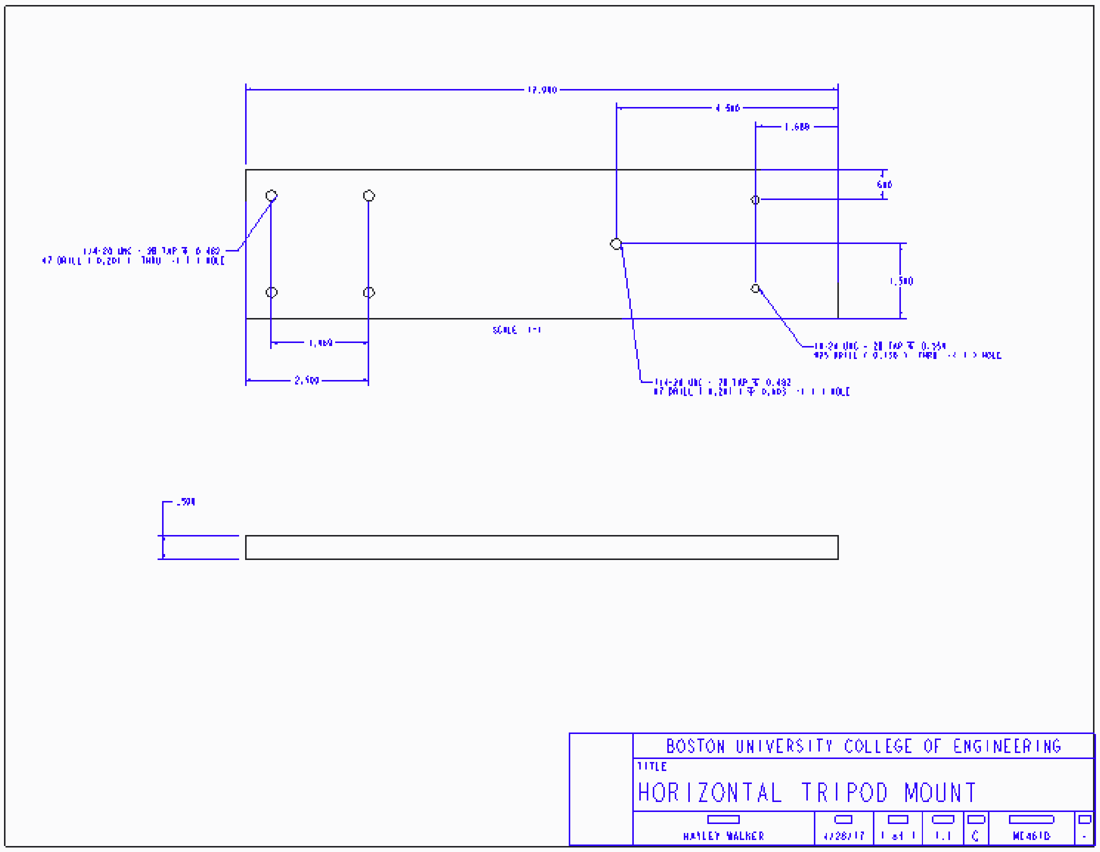
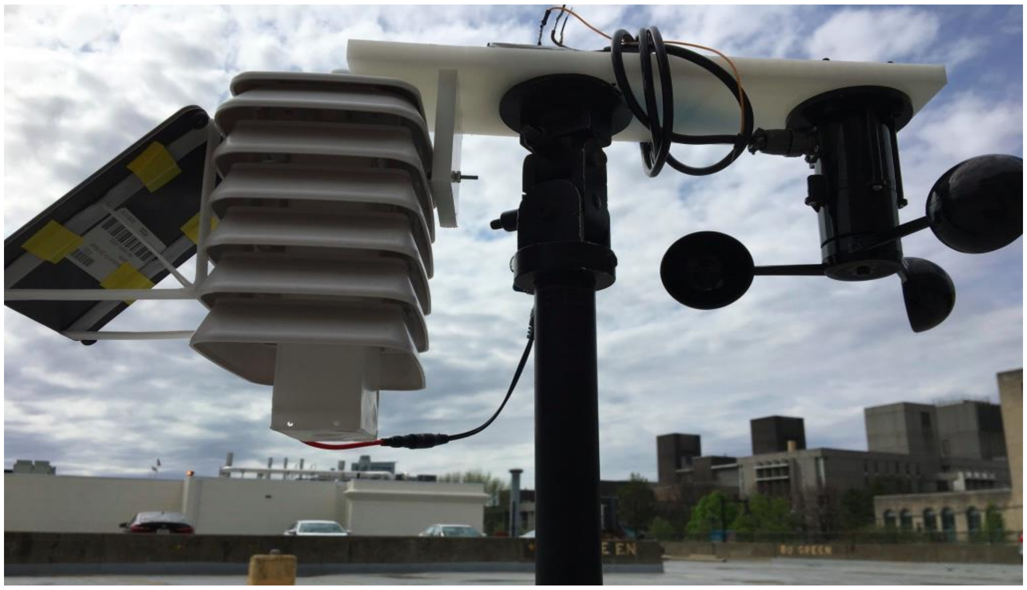

17.900
4.500
1.600
.600
1.500
1/4-20 UNC - 2B TAP ⊽ 0.482
#7 DRILL (0.201) THRU - (1) HOLE
1.400
2.500
SCALE 1:1
10-24 UNC - 2B TAP ⊽ 0.354
#25 DRILL (0.150) THRU - (2) HOLE
1/4-20 UNC - 2B TAP ⊽ 0.482
#7 DRILL (0.201) ⊽ 0.605 - (1) HOLE
.500
BOSTON UNIVERSITY COLLEGE OF ENGINEERING
TITLE
HORIZONTAL TRIPOD MOUNT
HAYLEY WALKER
4/28/17
1:1
C
ME461D
-

Figure 4. CAD Drawing of Mount.

The mount will look like this when complete:



BU GREEN

6 ANEMOMETER FROM SCRATCH

💲💲 ❓❓❓ ✂✂✂

5.1. SOME BACKGROUND

It is important to keep in mind that this section is a beta version, meaning that it is a work in progress. It is also important to note that while the components required (see BOM #2) are expensive added up, this system is slightly cheaper than the purchased anemometer option (hence the rating).

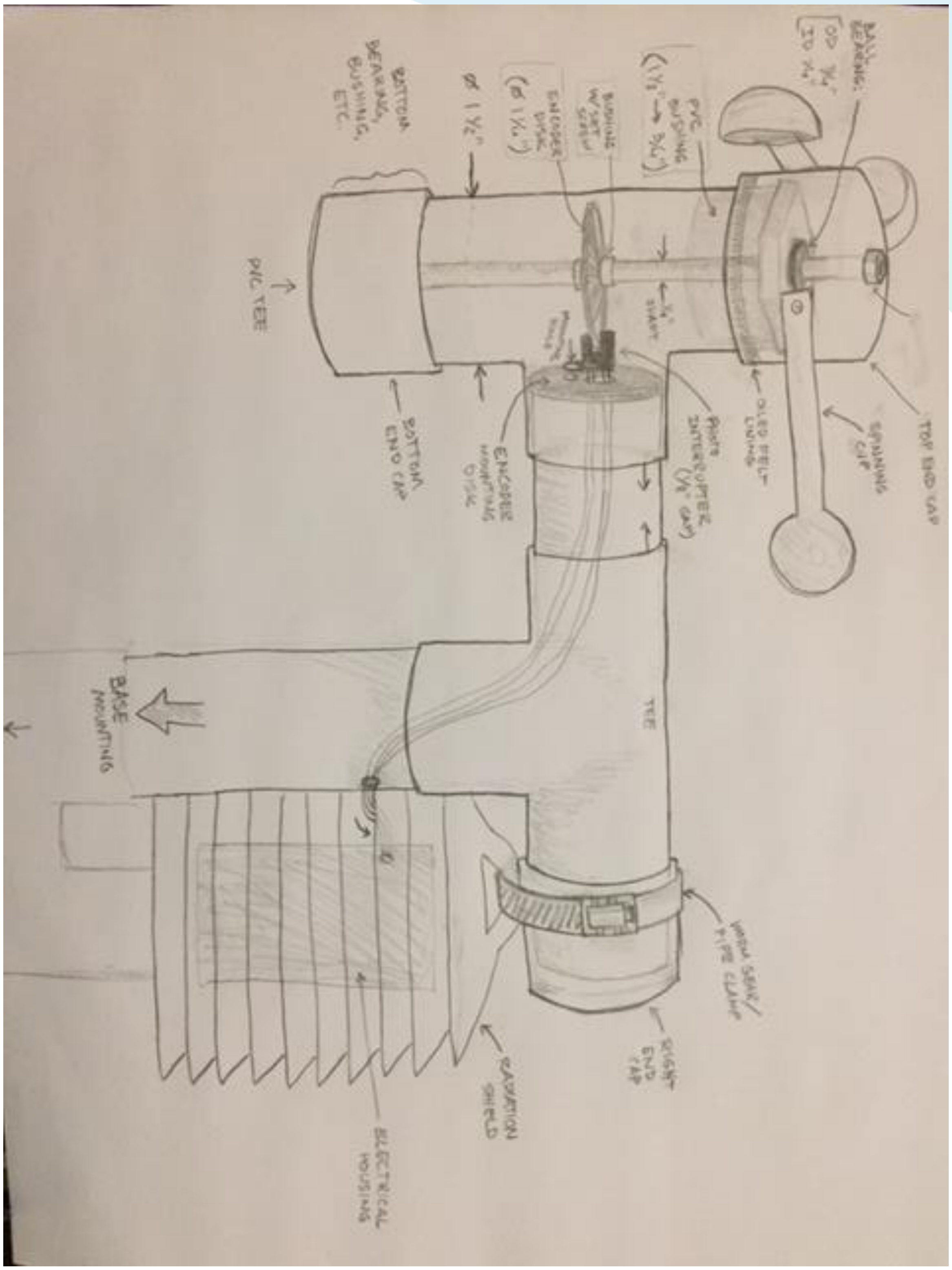
5.2 TOOLS

Some extra tools you'll need

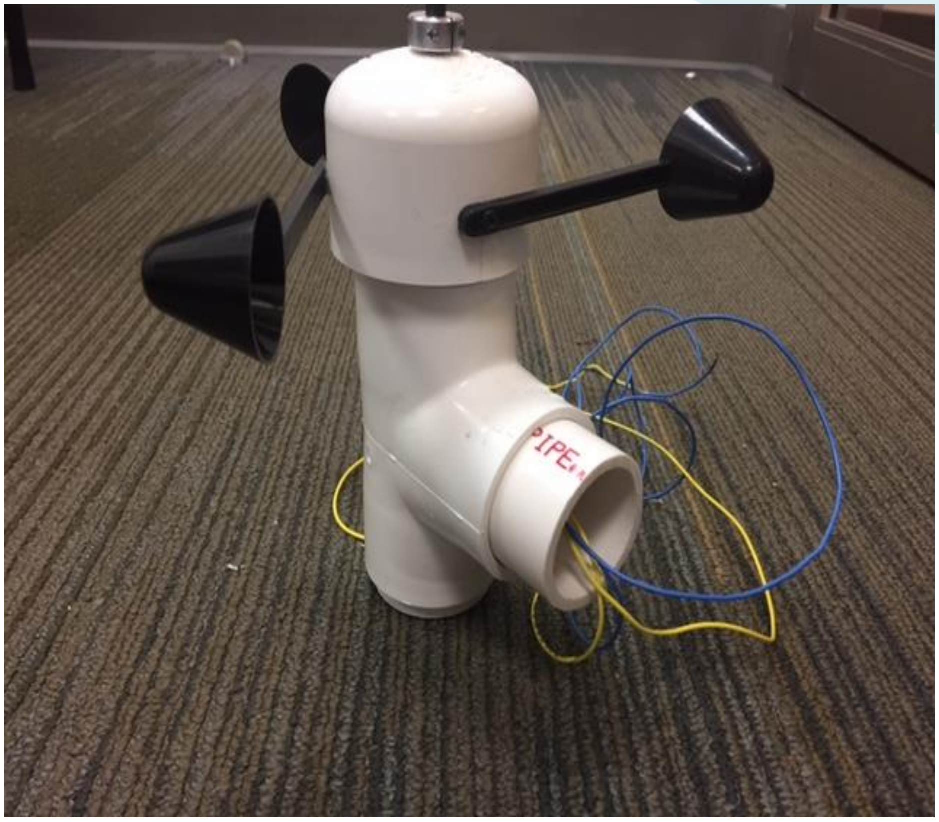
- ☐ Soldering iron and solder
- ☐ Screwdriver
- ☐ Hammer
- ☐ Hand drill and drill bits
- ☐ Freezer or table clamp
- ☐ Table saw
- ☐ X-Acto knife
- ☐ Allen wrench set

5.3 OVERALL DESIGN

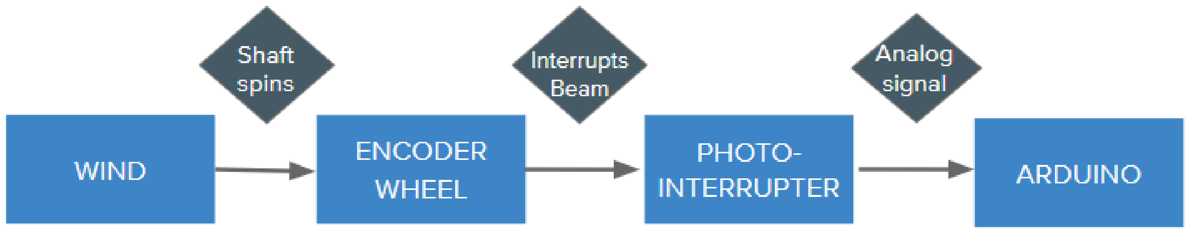
The final structure will look like this:


TOP END CAP
BALL BEARING:
OD 3/4"
ID 1/4"
SPINNING CUP
OILED FELT LINING
PVC BUSHING
(1 1/2" → 3/4")
1/4" SHAFT
PHOTO INTERRUPTER (1/8" GAP)
BUSHING W/ SET SCREW
ENCODER DISK
(ø 1 1/16")
MOUNTING HOLE
ø 1 1/2"
ENCODER MOUNTING DISK
BOTTOM END CAP
BOTTOM BEARING, BUSHING, ETC.
PVC TEE
TEE
WORM GEAR/PIPE CLAMP
RIGHT END CAP
RADIATION SHIELD
ELECTRICAL HOUSING
BASE MOUNTING

To give you an idea of the final product, ours ended up looking like this:

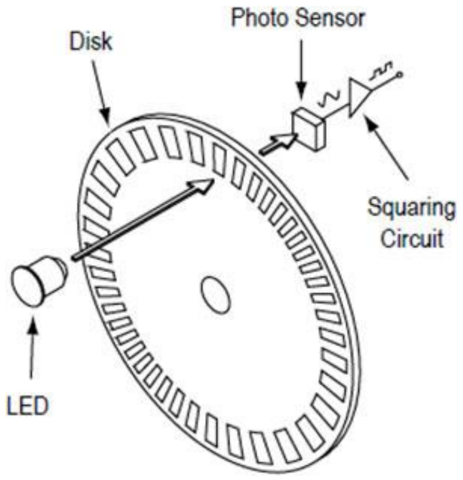


The main functions of the mechanism are as follows; the wind will catch the cups, which will spin the shaft. The shaft is attached to the encoder wheel, which will spin with it. As the wheel rotates, the photo interrupter beam will be interrupted, and will output an analog signal based on the amount of light the receiver can get. The signal will go to the Arduino, which outputs to the Serial Monitor. Luckily, we pre-calibrated this device for you, which will allow you to convert this output signal into a wind speed.



Let's get started.

5.4 ENCODER

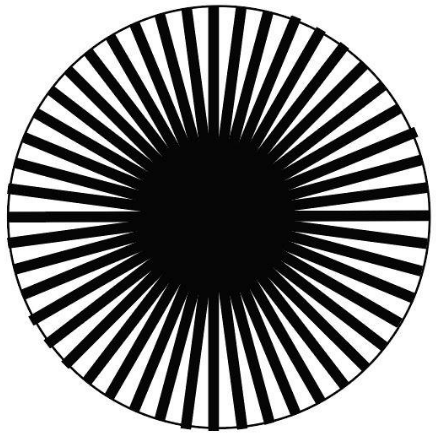


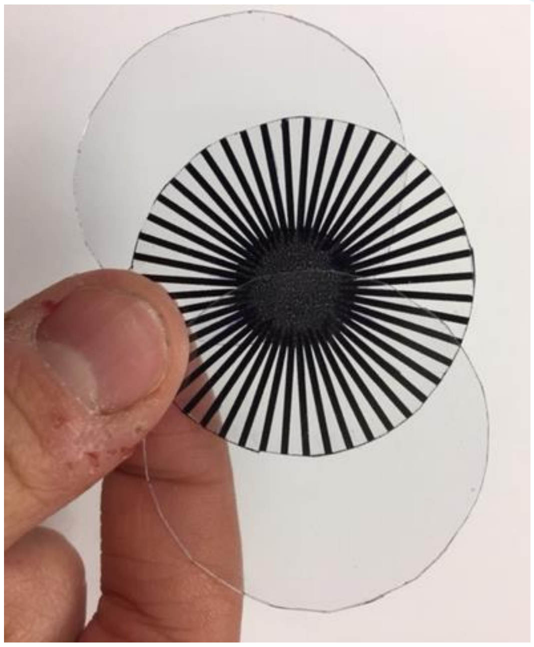
The heart of the FS anemometer is the encoder. An encoder is a mechanism to measure the speed at which something rotates. It usually does this by measuring the angle of a shaft relative to its initial angle, over time. An incremental encoder consists of a wheel with slits in it, and a sensor emitter and receiver on either side. Every time the sensor (called a photo interrupter) beam is blocked, a LOW signal is sent out to the Arduino. When the beam is not blocked (when it goes through a slit in the encoder wheel), a HIGH signal is sent to the Arduino. Here, we will make our own encoder to measure the speed at which a shaft rotates. The rotation is caused by the wind, similar to how wind moves the blades of a turbine in a circle. For more information about encoders, visit this site:

http://www.machinedesign.com/sensors/basics-rotary-encoders-overview-and-new-technologies-0

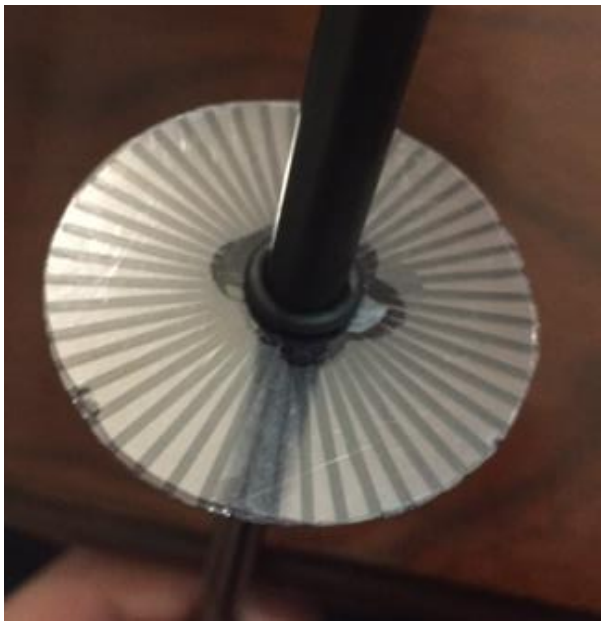
We designed the encoder wheel with 48 black stripes, like this:

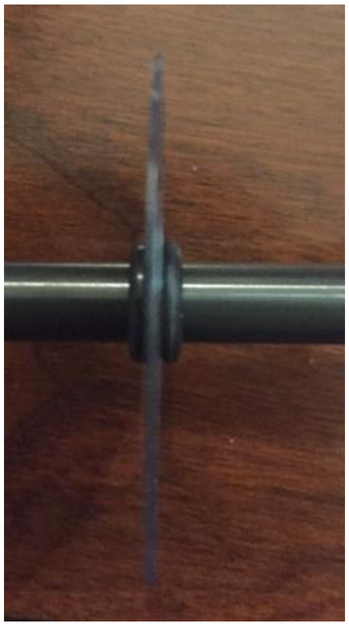
This beta Section 5 does not include an open source file with this design, but it is reproducible in any graphics program. The diameter is 1.5 inches.





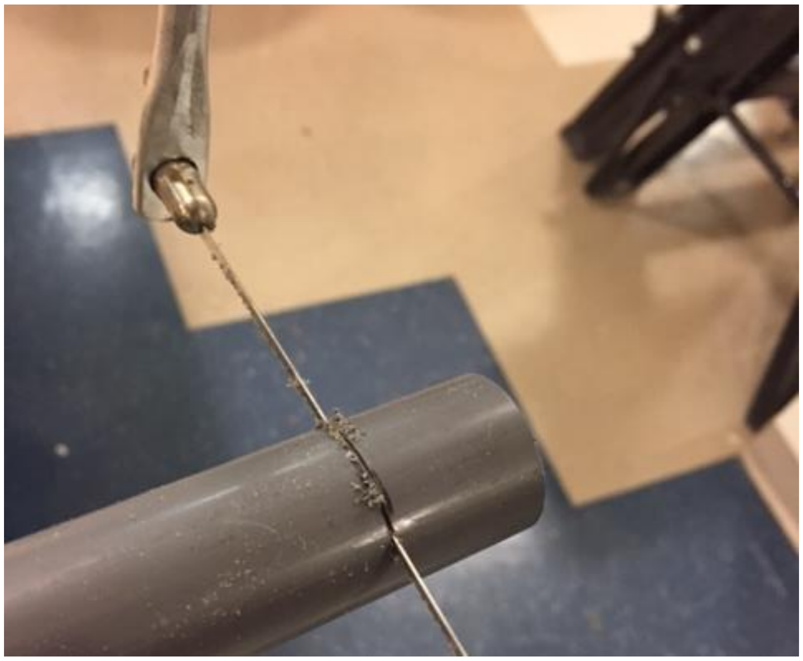
Take this file to a local print shop (we went to FedEx) and get it printed on a thick, clear projection sheet. Make sure it is the correct dimensions. Cut it out using an X-Acto knife (easier than scissors, and it won't bend your encoder wheel) or something similar. This sheet was a little thin, so we found some lamination plastic and cut it out in the same dimensions. Then we sandwiched the encoder wheel between the extra plastic circles, and cut a ¼" hole in all. We carefully super glued the edges of these together, as well as the inside edges of the hole. Now your encoder wheel is complete! Next, mount it onto the ¼' aluminum shaft. Slide it on 4.35 inches from one side of the shaft, and slide two O-rings onto either side. Push them snugly up against either side of the encoder wheel, and add a drop of superglue underneath. The finished assembly should look like this:





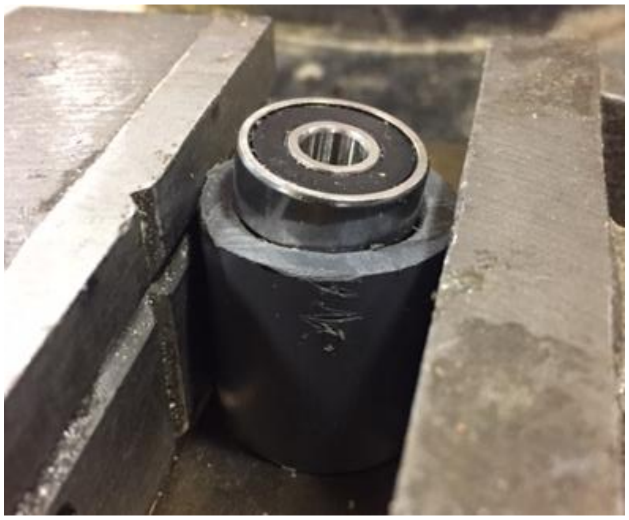
5.5 STRUCTURE

The main body of the anemometer is made from a PVC tee. Two bushings fit into either side, with a 1 1/2 " PVC plug on the bottom, and a 2" PVC cap on top. First, mount the bearing inside a section of small PVC pipe. Take the small pipe (this is ¾" schedule 40 PVC pipe) and saw a 1" section off. Make sure the table saw is positioned vertically.

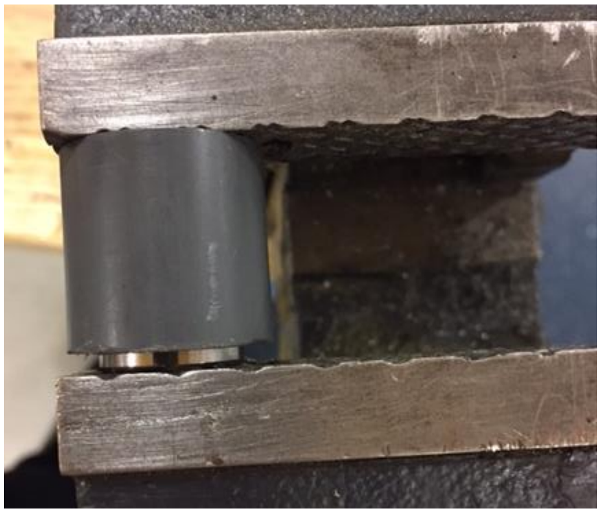


Next, press fit the bearing into the small pipe piece. This works best by positioning the pieces in a table clamp and slowing closing the clamp, as shown below. Another method is freezing the bearing for about an hour, to thermally contract the metal. The bearing should then fit into the pipe with a few gentle hammer taps (ONLY tap the outside ring of the bearing, not the inside ring—or it will be damaged).

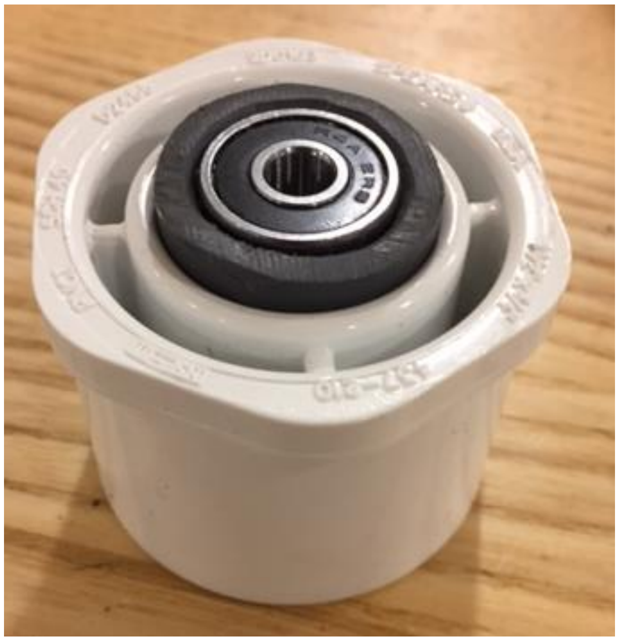
Position to tap bearing in with hammer



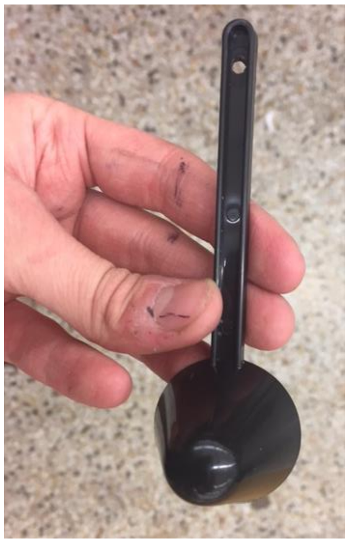
Position to press fit bearing with clamp



Next, fit the small pipe inside a bushing. If it doesn't slide on, use the same method outlined above to press fit the pipe and bearing into the bushing. When finished, it should look like this:

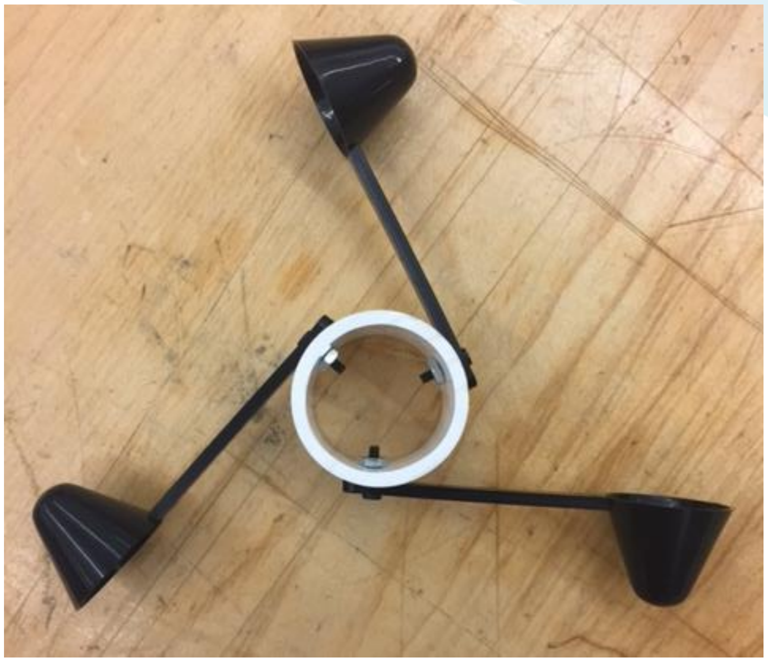


The following sections will be briefly explained, and will be detailed in future iterations of this instruction manual.



Next, construct the rotating cap. Drill three equally spaced holes in the side of the PVC cap, and drill one hole in each coffee scoop, as shown to the left.

Screw the coffee scoops onto the cap, 120 degrees apart. Attach nuts to the other sides and put a few drops of super glue behind each cup arm for good measure. It should look like the following when complete, but with a cap (and not open pipe):

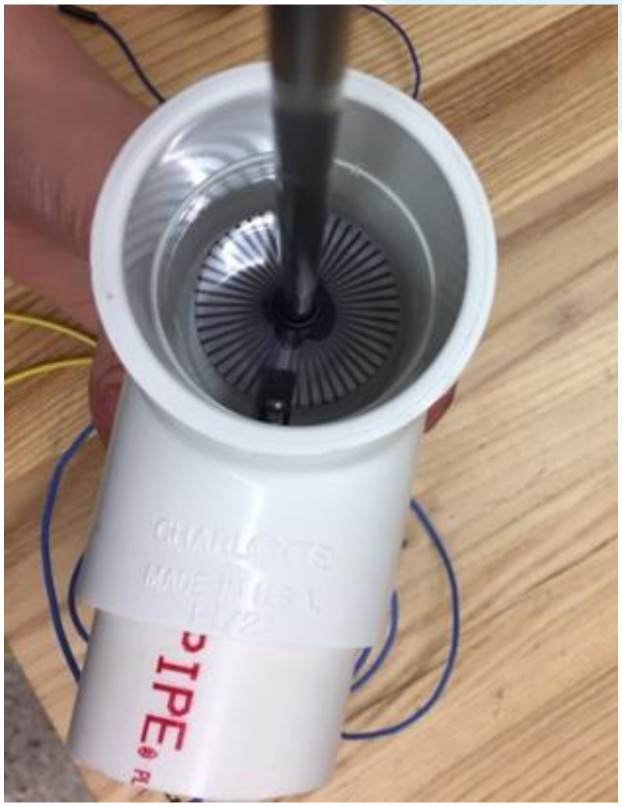


Next, cut a piece of larger 1 ½″ PVC pipe to 1 ft in length. Then drill a ¼″ hole into the top of the 2″ PVC cap.

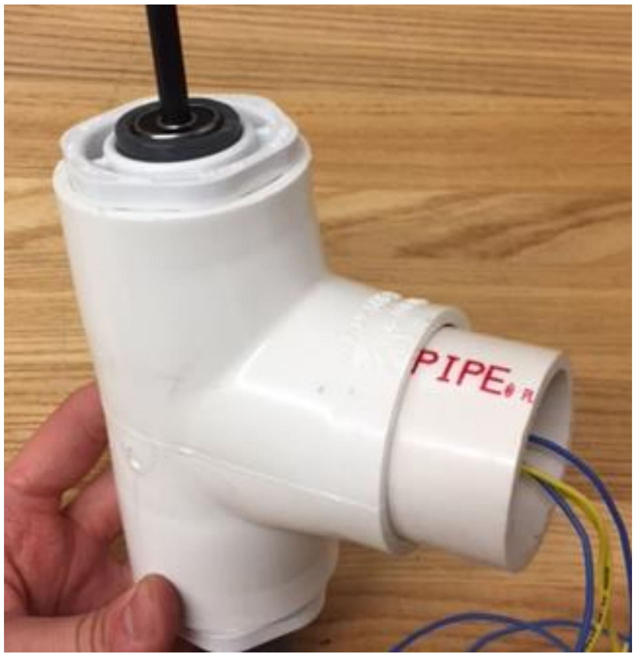
Next, assemble all components:

1. Stick the bushing/bearing assembly into the top part of a 1 ½″ PVC tee.
2. Slide the shaft and encoder assembly into the bearing and bushing assembly by pushing the shaft into the bearing. Push it through until 1.7 inches of the 6″ shaft sticks out of the top.
3. Stick a 1 ½″ PVC plug onto the bottom of the tee.
4. Attach one of the small shaft clamps to the top of the shaft that sticks out of the bearing, and tighten with an allen wrench.
5. Slide the 2″ PVC cap onto the aluminum shaft, so that it sits on the shaft clamp.
6. Attach the other shaft clamp onto the top of the PVC cap so that it is tight against the cap. Tighten it with an allen wrench. Make sure the cap does not rotate without the shaft.

The encoder/shaft assembly in the PVC tee looks like this:

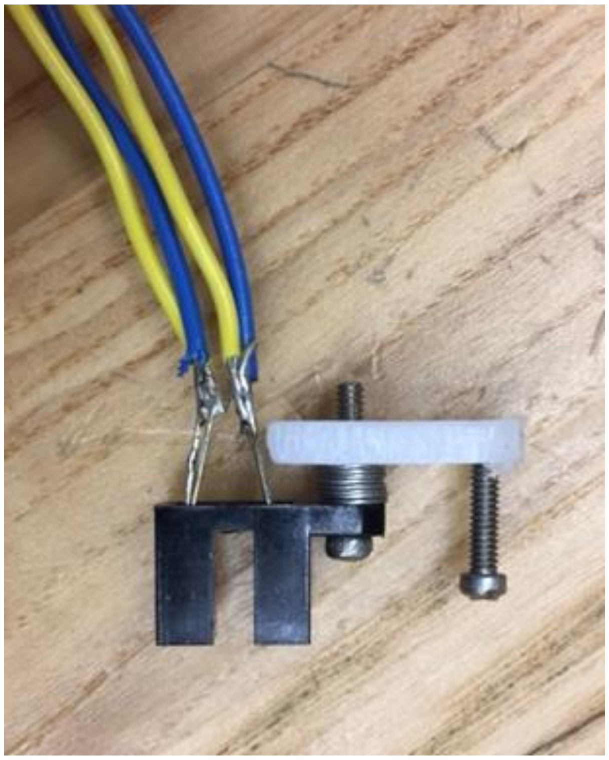


After step 4, the entire assembly should look like this:

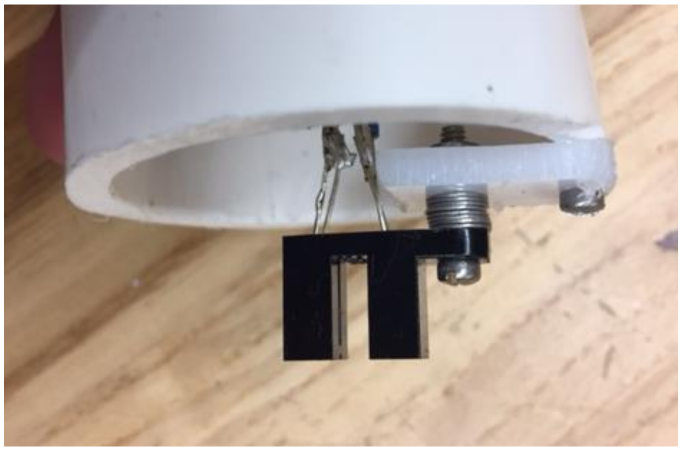


Next, make the photo interrupter and mount assembly.

Cut out a small piece of HDPE and screw two 3-48 holes in, one at the top and one at the bottom. Mount the photo interrupter as shown:



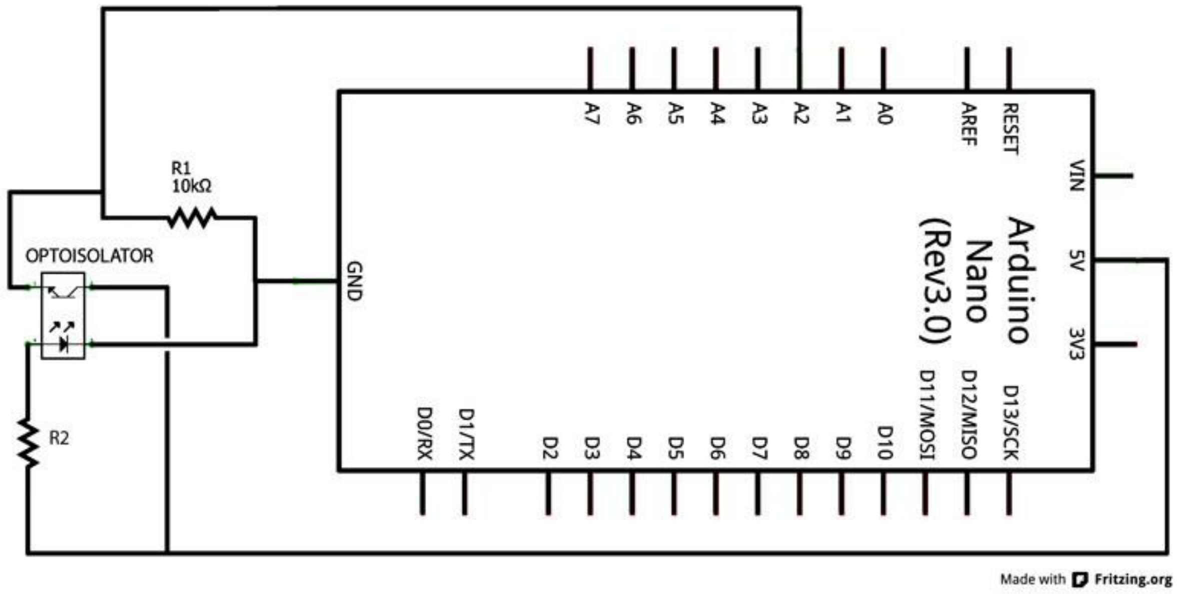
Solder four wires, one onto each photo interrupter pin, as shown above. Using different colors helps a lot with hardware. Next, drill a small hole perpendicular into the edge of the 1 ft. PVC piece cut earlier. Mount the component as shown, making sure the mount is vertical:



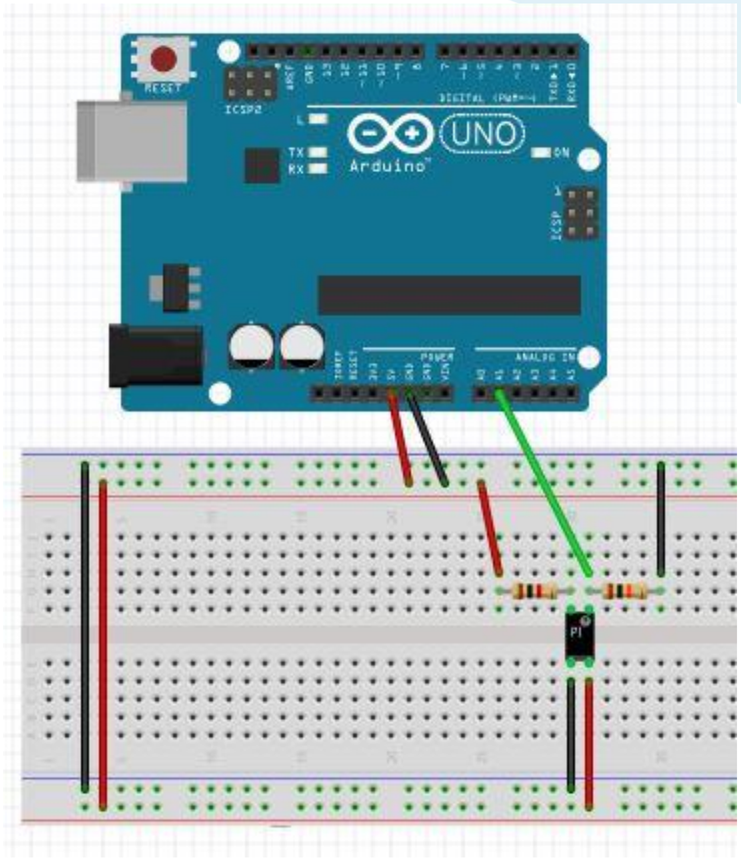
Stick the pipe piece into the perpendicular part of the tee, and push it in until it presses up against the tee ridge. This part is tricky, because you must make sure the encoder wheel and photo interrupter are aligned; shimmy the aluminum shaft up or down to make sure the encoder wheel fits in between the emitter and receiver of the photo interrupter. When the alignment is correct, squeeze a few drops of super glue onto the shaft where the bearing is attached, making sure not to glue the bearing racings together. Your complete anemometer structure should now look like the first pictures in this section.

5.6 HARDWARE AND SOFTWARE

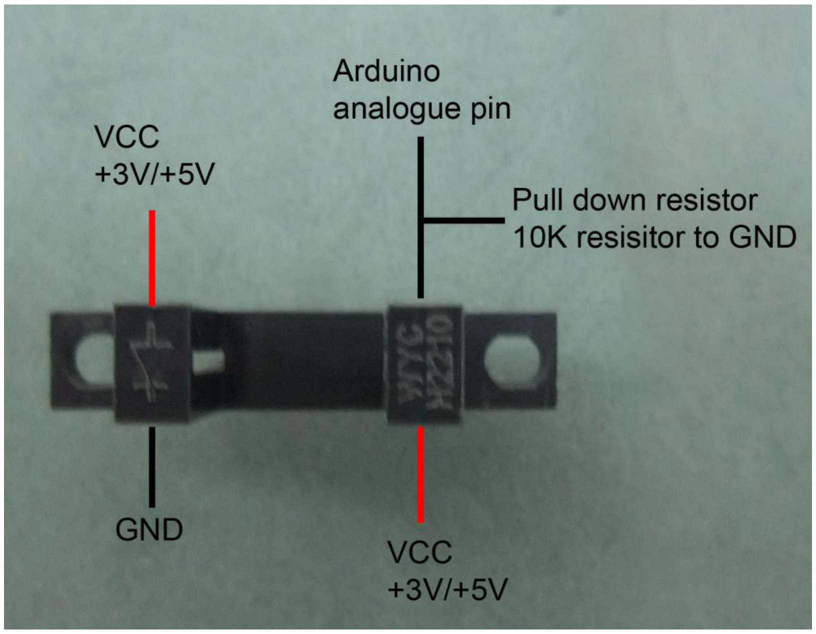
When an object passes between the emitter and detector of the photo interrupter, the beam is broken and the sensor outputs LOW. When the gap is free and the beam unbroken (the white slice of the encoder wheel) the output reads HIGH. While this is an analog output, the values read from the Serial Monitor will be higher or lower. Here is how to wire your system with the Arduino:



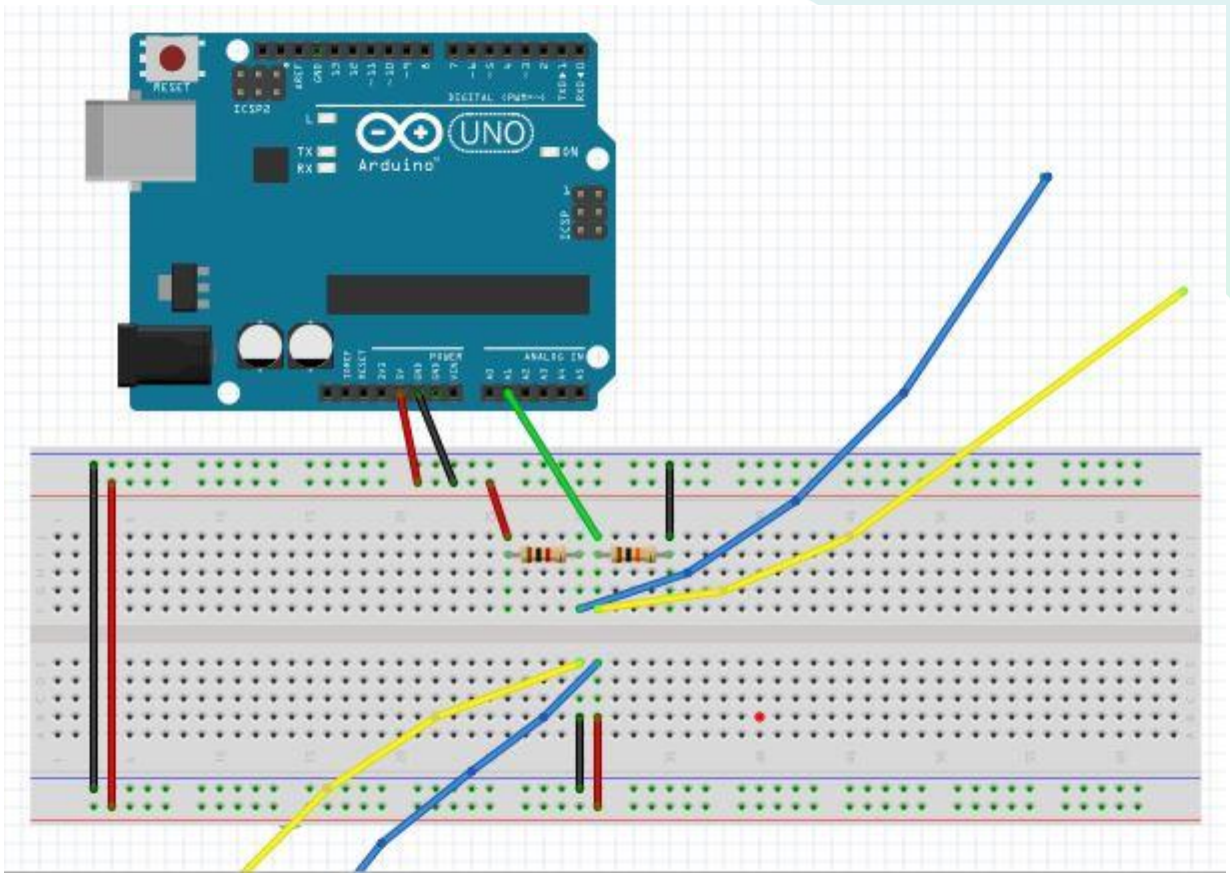
R2 is a 1KΩ resistor, and R1 is a 10KΩ resistor. For a more intuitive schematic of the setup, refer to the picture below:



As depicted above, try wiring your breadboard with a spare photo interrupter first, to test the system. Here is how the photo interrupter is connected to the rest of the system:



Once you are comfortable with this setup, replace the sensor with the wires leading out of the anemometer structure, so that is looks more like this:



Test for functionality by copy pasting the code below into the Arduino IDE. This code outputs an analog value into the Serial Monitor from the photo interrupter input.

```
/* Analog Read Interrupter
* ------------------------
*/

int PhotoPin = A2; // select the input pin for the interrupter
int val = 0; // variable to store the value coming from the sensor

void setup()
{
    Serial.begin(115200); // set up Serial library at 9600 bps
}

void loop()
```

```
{
    val = analogRead(PhotoPin); // read the value from the sensor
    //if (val < 300){
  // Serial.println("BLOCKED");
 // }
  //else {
    Serial.println(val); // print the sensor value to the serial monitor
    delay(1);
//   }

}
```

Try spinning the shaft and see what happens to the outputs in the Serial Monitor. The homemade anemometer output, as seen when this code is implemented, is random analog numbers spit out from the photo interrupter: meaningless without calibration. Luckily, this device was calibrated in a wind tunnel. If any alternative designs or structural components were integrated in your design (different shaped cups, etc), it is recommended you make your own calibration curve by blowing air from a fan at various speeds, and using a handheld anemometer (in BOM 2) to correlate wind speed with encoder output values. For now, our calibration is:

$$y = 0.03x + 0.6622$$

In which y is wind speed in m/s, and x is anemometer rotation speed in rad/s.
However, this is not complete. To use this calibration, we must convert the analog output given to us in the serial monitor to a rotation speed in radians per second. This is done by finding the frequency of HIGH point output, with the following code:

```
/* Analog Read Interrupter
* ------------------------
WITH CALIBRATION
*/

int PhotoPin = A2; // select the input pin for the interrupter
int HP = 0; //High Points in photo interrupt output
int outputs[500];
int val = 0;

void setup() {
Serial.begin(115200); // set up Serial library at 115200 bps
}
```

```
void loop() {
 //GET DATA (0.5 sec)
  //take a 0.5 second sample of data, load data into a vector to store
  for (int i = 0; i < 500; i++){
    int val = analogRead(PhotoPin); // read the value from the sensor
    outputs[i] = val;
    delay(1);
  }
  //FIND MAX VALUES IN DATA, GET HP VALUE
  for (int j = 1; j < 500; j++) {
    //find the high points; if a point is greater than the points
    //around it and if its higher than 950, its a HP.
     if (outputs[j] > 950 && outputs[j] > outputs[j-1] && outputs[j] >
outputs[j+1]){
       HP++;
     }
  }
  //CALCULATE AND PRINT VELOCITY
  int x = HP*0.131;
  int rads = x/0.5; //[rad/s]
  // v = sqrt(2gh), but use the calibration v = 0.03(rads) + 0.6622
  int velocity = 0.03*rads + 0.6622;
    Serial.println(val); // print the sensor value to the serial monitor
    Serial.println(velocity);
    delay(6000); //delay 1 min before taking another 10 s sample
}
```

NOTE: Unfortunately this code is buggy. You may need to make some alterations to make it work (get an instructor to help).

And that's it! If you got this far, you now have a working standalone anemometer. Further iterations of this manual will include separate mounting instructions, as well as hardware and software integration with the rest of the system.

5.7 Additional Resources:

Here are some additional resources we found helpful when this part of the project. They may be of help to you too.

These are a home-built anemometers similar to our own:

http://www.hackersbench.com/Projects/anemometer/anemometer3.html http://www.raphnet.net/divers/anemometre/anemometer_en.php

This explains more about how to wire a photo interrupter:

http://www.martyncurrey.com/connecting-an-photo-interrupter-to-an-arduino/